RAND

An Approach to Replicated Databases for Robust Command and Control

Iris Kameny

Arroyo Center
National Defense Research Institute

19961029 076

DTIC QUALITY INSPECTED 1

The research described in this report was conducted within two RAND federally funded research and development centers: The Arroyo Center sponsored by the United States Army, Contract MDA903-91-C-0006; and by the Advanced Research Projects Agency. The ARPA research was conducted in RAND's National Defense Research Institute, supported by the Office of the Secretary of Defense, the Joint Staff, and the defense agencies, Contract No. MDA903-90-C-0004.

ISBN: 0-8330-2338-1

© Copyright 1995 RAND

All rights reserved. No part of this book may be reproduced in any form by any electronic or mechanical means (including photocopying, recording, or information storage and retrieval) without permission in writing from RAND.

RAND is a nonprofit institution that helps improve public policy through research and analysis. RAND's publications do not necessarily reflect the opinions or policies of its research sponsors.

Published 1995 by RAND
1700 Main Street, P.O. Box 2138, Santa Monica, CA 90407-2138
RAND URL: http://www.rand.org/
To order RAND documents or to obtain additional information, contact Distribution Services: Telephone: (310) 451-7002; Fax: (310) 451-6915; Internet: order@rand.org

LIBRARY OF CONGRESS
FEB 22 1996
GIFT

RAND

An Approach to Replicated Databases for Robust Command and Control

Iris Kameny

Prepared for the
United States Army
Advanced Research Projects Agency

Arroyo Center
National Defense Research Institute

Approved for public release; distribution unlimited

Preface

This report presents an approach to making future command and control data more timely and robust through the use of replicated distributed data management techniques. The types of command and control data that we address in this report include "situational awareness" data needed by Army tactical commanders from platoon through corps, including mission plan and progress, represented as machine-processable operation orders (OPORDs), enemy situation, and friendly situation and status.

The concepts presented here were developed in answer to common objectives in two projects, a Distributed Databases project for the Advanced Research Projects Agency (ARPA) of the Department of Defense (DoD), and a project addressing the Management of Information for Joint and Combined Operations for the Army Office of the Director of Information Systems for Command, Control, Communications, and Computers (ODISC4).

The report should be of interest to C2 operators and technologists concerned with digitization of the battlefield, tactical command and control, replicated distributed databases, and handling of asynchronous data on the battlefield.

The research sponsored by ARPA was conducted within the Acquisition and Technology Policy Center of RAND's National Defense Research Institute (NDRI). NDRI is a federally funded research and development center sponsored by the Office of the Secretary of Defense, the Joint Staff, and the defense agencies.

The research sponsored by the Army Office of the Director of Information Systems for Command, Control, Communications, and Computers was conducted in the Force Development and Technology Program of the Arroyo Center.

Contents

Preface iii

Figures and Table vii

Summary ix

Acknowledgments xiii

Acronyms xv

1. INTRODUCTION AND BACKGROUND 1
   - Objectives 1
     - ARPA Research Objectives 1
     - Army Research Objectives 2
   - Motivation 2
     - Recent Army 21st Century Endeavors 3
     - Inadequacy of Threat Assessment 4
     - Insights from Operation Desert Storm 4
   - Focus of Study 6
   - Technology Areas 9
     - Management of Distributed, Replicated Data 9
     - Database Consistency 10
     - Convergence Toward Consistency 11
   - Related Efforts 11

2. APPROACH TO REPLICATED DATABASES FOR ROBUST COMMAND AND CONTROL 13
   - Design Concepts 13
     - Design Overview 13
     - Command and Control Data-Handling Principles 14
     - Historical Data Management 15
     - Knowledge Managers and Mediators for Army Unit 15
     - Design Rationale for Data Distribution to C2 Units 18
     - Robustness 19
     - Transaction Management 19
     - Synchronization Through Use of Sequence Numbers and Liveness Reports 21
     - Maintaining Consistency at the Local Node 23
     - Types of Messages 24
     - Basing C2 Data on DoD Data Model and DoD Data Standards 26
     - Categories of Data 28
   - Examples of Handling C2 Events That Support Situational Awareness 30
     - Example One: Position Location Change of a Friendly Unit 31
     - Example Two: A Situation Report 34
     - Example Three: Change in Operation Orders (OPORDs) 38
     - Example Four: Handling of Intelligence Reports 42
   - Discussion of the Examples 46

3. RESEARCH ISSUES AND FUTURE DIRECTIONS 49
Object-Oriented Technology Combined with Historical Data Management 50
Replication Through Use of Knowledge Managers and Mediators for Command and Control 51
Handling of Local Node Long Transactions 53
Inference, Estimation, Temporal Logic, and Reasoning 53
Handling of Intelligence Data and Security Issues 55
Issues for Future Exploration and Development 56
Military Issues 56
Technical Issues 57

Appendix: Transaction Processing and Concurrency Control 59

References 63

Figures

1. Concept of Nonhierarchical Information Carousels in Force XXI . . . 8
2. Corps Through Platoon Command Hierarchy 14
3. Example Distribution of Knowledge Managers and Mediators for Replicated C2 Databases 17
4. Fact Types and Some Relations 29

Table

1. Tactical Data Communications Conceptual Framework 7

Summary

This report presents an approach to making future command and control data more timely and robust through the use of replicated distributed data management techniques. The types of command and control data that are addressed in this report include "situational awareness" data needed by Army tactical commanders from platoon through corps, including mission plan and progress, represented as machine-processable operation orders (OPORDs), enemy situation, and friendly situation and status.

The concepts presented in this report were developed in answer to common objectives in two projects, a Distributed Databases project for the Advanced Research Project Office (ARPA) of the Department of Defense (DoD), and a project addressing the Management of Information for Joint and Combined Operations for the Army Office of the Director of Information Systems for Command, Control, Communications, and Computers (ODISC4).

Motivation

The report describes a conceptual solution to the problem of giving warfighters near-real-time common pictures of the battlefield for situational awareness in spite of communication delays due to jamming and congestion, network partitioning, node failures, and other hostile actions that might interfere with the rapid dissemination of data.

The ARPA sponsor asked us to explore a replicated database approach, assuming future technology including gigabyte networks and memories, terabyte secondary and archival storage systems, and multiprocessor and distributed system architectures, to see if these technology advances could be used advantageously to support C2 robustness. The main issue is the conflict between the need for data in near-real-time everywhere versus the need for data consistency everywhere. Delays in transmission (even normal transmission operations) will mean that replicated data cannot be guaranteed to arrive everywhere at the same time. Given that timely availability of data is extremely critical to C2, the problem becomes one of managing data inconsistencies across C2 units in such a way that distributed data will eventually converge toward consistency at all the C2 units.

The objective of the Army study effort was to support the current Army future-looking endeavors such as Army Force XXI, the Army Digitization Office (ADO), the Army Battle Command System (ABCS), and the Army Enterprise Strategy (AES).

The following quotation is directly related to the issues of this study: ". . . despite advances in information technology, commanders, leaders, and soldiers will never have perfect knowledge of the operational situation surrounding them. Yet, due to the pace and complexity of future battle, commanders, more so than in the past, must accept uncertainty and not hesitate to act instead of waiting for more analysis or information." [TRADOC PAM 525-5, 1994, p. 3–4.]

Approach

The focus of this study was to examine data asynchrony assuming full replication of dynamic command data from battalion to corps. Partially replicated data were assumed for units below battalion and the study does not address how to achieve consistency, timeliness, and robustness when data are only partially replicated. The study does not address communication technology and media, or computer and security technologies, except for noting the context in which topics may become issues. For simplicity, it has treated communications between units as point-to-point communications, though other alternatives are possible and are addressed in Section 1.

We discuss two different ways the Army sees information being managed and provided on the battlefield. One is through nonhierarchical information carousels (organized according to functional areas such as logistics or intelligence) and the other through hierarchical and horizontal distribution of dynamic combat data. Our study is focused on the use of data replication to manage and provide dynamic combat data, though we provide an information carousel example in the management of intelligence data. Our design supporting the replication of dynamic combat data requires that a Knowledge Manager for a C2 combat unit (KMC) exist for each C2 unit (e.g., battalion, brigade, division, and corps). The KMC manages all data unique to that unit through the Army Common Operating Environment (COE) software augmented by any unit/mission-specific software including decision-aiding software. The KMC can be thought of as an automated assistant to the unit commander or staff for carrying out the unit's mission. Changes to the unit's database are initiated only through the unit's KMC. In addition to managing the unit's data, the KMC is responsible for managing the propagation of the unit's data to every other C2 unit. The replications are handled through Mediator(s) for the C2 unit (MDC) at

every other C2 unit. The KMC and MDCs for a given C2 unit must have identical software. Any changes to a unit's KMC management software needs to be synchronized with software changes to its MDCs at every other C2 unit. Knowledge managers and mediators are discussed in detail in Section 2.

Our approach uses a weak data synchronization mechanism between nodes, thereby allowing decisions to be made and actions taken based on inconsistent data and inferences. It requires management of historical data, keeping track of actions that are based on the use of possibly inconsistent estimated data, and methods for propagating changes when consistency is later achieved. The synchronization method uses sequence numbers on messages, and KMC aliveness messages and replies by MDCs are used to inform the KMC as to the state of the KMC's database on each remote node.

The overall result is support for timely data that at any point in time may be inconsistent but will converge over time toward a consistency or degree of correctness with respect to the real world. This approach allows the commander and his staff to use the near-real-time data immediately in decisionmaking and execution while also providing support for tracking data dependencies in decisions based on hypothetical data and propagating later critical data changes to the relevant actors. If data are late or missing, they may be estimated or inferred, labeled as hypothetical, and used in the KMC decisionmaking process.

The use of historical data and total data replication could support a very robust C2 system. Essentially, a commander and his staff could operate through their KMC at the node they select, which by preference would most likely be their command post node. By using replication, a commander whose node is in trouble could choose to move his KMC (and decisionmaking) to another node.

Examples of Handling C2 Events That Support Situational Awareness

For purposes of illustration this report walks through, at a high conceptual level, four events necessary to support situational awareness. These events indicate the kinds of system activities required to pass information horizontally and vertically for near-real-time processing concurrently throughout the entire system. They are (1) showing the ripple effect when a platoon has a location change that exceeds a threshold; (2) demonstrating the generation of a division-level Situation Report (SITREP) when one of its brigade's SITREPs is missing and has to be estimated; (3) illustrating how a change in a platoon's ability to perform its OPORD would ripple upward to corps and the dissemination of changed OPORDs from corps to platoon; and (4) showing the propagation of an

intelligence report from platoon upward (i.e., due to sighting and engaging the enemy) and from corps downward.

Issues for Future Exploration

We identified military and technical issues needing more attention in the future.

Military Issues for Future Exploration

1. Need to bring together Joint Force and Army current and future doctrine, operational requirements, and future visions for battle management to interact with and drive proposed technological solutions.
2. Need to better understand concept of carousels of data shown in TRADOC PAM 525–5 (see Figure 1 in Section 1).
3. Need to explore doctrine to allow units to change their actions based on current situational assessment without or before receiving orders from their commanding unit.
4. Need to explore the effect of voice communications: If critical information is exchanged only by voice and not entered into the data system, then inferencing techniques including simulation will be limited.
5. Need to address data security issues including (a) risk at battalion and below, (b) dealing with classified data that have been downgraded to protect source; and (c) data aggregation risk posed by extensive replication.

Technical Issues for Future Exploration

1. Explore extensions to Object Oriented Database Management Systems (OODBMS) technology to support application-defined relationships and historical data.
2. Develop research concepts for use of long transactions and complex transactions to support the warfighter-decisionmaker.
3. Research use of knowledge management techniques for synchronization of data through propagation of critical changes.
4. Explore use of simulation to estimate behavior of other C2 nodes.
5. Explore categorization of types of data.
6. Investigate issues in use of DoD data standards based on relational technology with respect to moving to OODBMS technology.
7. Further explore robustness.

Acknowledgments

The author would like to thank Stephanie Cammarata, Phillip Feldman, and Gaylord Huth for their informal review of the report, John Bondanella for a formal review of the report, Sam Chamberlain for a review of the sections discussing the Army Research Laboratory's Information Distribution Technology research program, and Ruth Eagle and Linda Quicker for preparing the report for publication.

Acronyms

2PL	Two-Phase Locking
ABCS	Army Battle Command System
ACOE	Army Common Operating Environment
ADA	Air Defense Artillery
ADO	Army Digitization Office
AES	Army Enterprise Strategy
ARL	Army Research Laboratory
ARPA	Advanced Research Projects Agency
ATCCS	Army Tactical Command and Control System
AWIS	Army WWMCCS Information System
BFA	Battlefield Functional Area
C2	Command and Control
C4	Command, Control, Communications, and Computers
C4I	Command, Control, Communications, Computers, and Intelligence
C4RDP	Command, Control, Communications, Computers Requirements Definition Program
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CCIR	Commander's Critical Information Requirements
CCS	Command and Control Systems
CECOM	Communications Electronics Command
CFSW	Center for Software
CINC	Commander-in-Chief
CNR	Combat Net Radio
COA	Courses of Action

COE	Common Operating Environment
CONUS	Continental United States
COTS	Commercial Off-The-Shelf
CP	Command Post
DBMS	Data Base Management System
DDBMS	Distributed Data Base Management System
DDRS	Defense Data Repository System
DIS	Distributed Interactive Simulation
DISA	Defense Information Systems Agency
DoD	Department of Defense
DSS	Decision Support System
ECA	Event-Condition-Action
FDAd	Functional Data Administrator
FRAGO	Fragment Order
GCCS	Global Command and Control System
GPS	Global Positioning System
HCI	Human Computer Interface
HQ	Headquarters
IDEF0	Integration Definition Language for Function Modeling
IDEF1X	Integration Definition Language for Information Modeling
IDT	Information Distribution Technology
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
JIEO	Joint Interoperability Engineering Organization
JTF	Joint Task Force
KMC	Knowledge Manager for C2
KMI	Knowledge Manager for Intelligence
LAN	Local Area Network

MCEB	Military Communications Electronic Board
MCS	Maneuver Control System
MDC	Mediator for C2
MDI	Mediator for Intelligence
MG	Major General
NATO	North Atlantic Treaty Organization
NBC	Nuclear, Biological, and Chemical
NCA	National Command Authority
NDRI	National Defense Research Institute
ODISC4	Office of the Director of Information Systems for Command, Control, Communications, and Computers
ODS	Operation Desert Storm
OODBMS	Object Oriented Data Base Management System
OPLAN	Operational Plan
OPORD	Operational Order
PAM	Pamphlet
PEO	Program Executive Office
PM/CHS	Program Manager/Common Hardware Software
POS/NAV	Position/Navigation
SDE	Standard Data Element
SITREP	Situation Report
TAFIM	Technical Architecture Framework for Information Management
TO	Timestamp Ordering
TOE	Table of Organization and Equipment
TRM	Technical Reference Model
USMTF	United States Message Text Format
WWMCCS	World Wide Military Command and Control System

1. Introduction and Background

Objectives

This report presents an approach to making future command and control data more timely and robust through the use of replicated distributed data management techniques. The types of command and control data that we address in this report include "situational awareness" data needed by Army tactical commanders from platoon through corps, including mission plan and progress, represented as machine-processable operation orders (OPORDs), enemy situation, and friendly situation and status.

The concepts presented here were developed in answer to common objectives in two projects, a Distributed Databases project for the Advanced Research Project Agency (ARPA) of the Department of Defense (DoD), and a project addressing the Management of Information for Joint and Combined Operations for the Army Office of the Director of Information Systems for Command, Control, Communications, and Computers (ODISC4).

ARPA Research Objectives

The objective of the ARPA distributed databases effort was to develop distributed data management methodology to ensure some degree of "consistency" in replicated distributed near-real-time battlefield databases. In this context, the requirement for "consistency" must (1) assume battlefield communication delays and node failures and (2) permit the use of near-real-time, possibly inconsistent, data in making timely decisions and taking real-world actions. For this study we distinguish near-real-time as being less constrained than real-time data as used in a process control system where the data processing must be accomplished within a fixed time interval. The ARPA vision is of the near future when advanced technology such as gigabyte networks and memories, terabyte secondary and archival storage systems, and multiprocessor and distributed system architectures will make it possible to replicate data at all levels of command in near-real-time to serve multiple decisionmakers and at the same time ensure robustness. The technical problem to be addressed was data consistency across replications, especially in the face of jamming. The issue was how to assure, after the jamming ceased, convergence toward synchronization of

information at different nodes rather than divergence. An additional use, if the methodology proved feasible, would be to give the United States the ability to jam frequencies used by both hostile and friendly forces on the battlefield while not seriously impeding U.S. and friendly command and control functions.

Army Research Objectives

The objective of the Army study effort was to support Army future-looking endeavors such as the Army Digitization Office (ADO), the Army Battle Command System (ABCS), and the Army Enterprise Strategy (AES). The study has begun to conceptualize and analyze ways to provide information management to allow the Army to support the full spectrum of combat and noncombat contingency operations including participation in joint and coalition forces. The Army study focus has been on brigade and below, and on the distributed data/information/knowledge management requirements to meet battlefield needs in the year 2010. Below we briefly describe these efforts, which have served as a prime motivation for this Army study project.

In the remainder of Section 1, we first discuss some efforts that motivated this study, we describe the focus of the study, and we conclude with a discussion of the main technology areas of concern. In Section 2, we present an approach for addressing these problems. In Section 3, we discuss research issues and future directions.

Motivation

The following quotation from TRADOC PAM 525–5 is directly related to the issues of this study: ". . . despite advances in information technology, commanders, leaders, and soldiers will never have perfect knowledge of the operational situation surrounding them. Yet, due to the pace and complexity of future battle, commanders, more so than in the past, must accept uncertainty and not hesitate to act instead of waiting for more analysis or information." [TRADOC PAM 525–5, 1994, pp. 3–4.] A voluminous amount of information will be needed on the battlefield, which will require intelligent processing and distribution of critical data. Decisionmaking systems will be used for routine battle functions. They will operate in accordance with the commander's predetermined decisionmaking criteria. More advanced decision aids will also be needed.

Recent Army 21st Century Endeavors

TRADOC PAM 525–5 defines five important characteristics of the 21st century Army. They are doctrinal flexibility, strategic mobility, tailorability and modularity, joint and multinational connectivity, and versatility to function in war and operations other than war. It recognizes the need for hierarchical (e.g., command structure) and nonhierarchical (e.g., intelligence and logistics) internetted processes, horizontal integration of battlefield functions, and centralized and decentralized command and control. Knowledge-based land warfare through horizontal and vertical data distribution will provide real-time situational awareness. This means that commanders at every level will share a common, relevant picture of the battlefield scaled to their level of interest and tailored to their special needs. This will give maneuver, combat support, and combat service support leaders the means to visualize how they will execute in harmony. It will give subordinates as much information as their commanders, and individual soldiers will be empowered for independent action.

In the May 1994 issue of *Army*, BG Joseph E. Oder, the leader of the Army Digitization Special Task Force, defined digitization as "the application of information technologies to acquire, exchange and employ digital information throughout the Battlespace, tailored to the needs of the decider (commander), shooter, and supporter, allowing each to maintain a clear and accurate vision of the Battlespace necessary to support planning and execution." [Oder, 1994]. The ADO has since been established as the focal office to carry out digitization of the battlefield. It is directed by MG Rigby.

The ABCS [USA ABCS, 1994] includes all of the strategic, operational, and tactical Army battlefield systems from the Army World Wide Military Command and Control System (WWMCCS) Information System (AWIS), through the theater-level systems and the Army tactical systems, i.e., Army Tactical Command and Control System (ATCCS). It will implement the horizontal and vertical internetted processing described in Force XXI to provide distributed situational awareness of the battlefield. It includes an Army version of the Global Command and Control System (GCCS) and the Common Operating Environment (COE), and it will track Army C4 system requirements in the C4 Requirements Definition Program (C4RDP) database.

The Army Enterprise Strategy [USA Enterprise, 1994] integrates both current Army doctrine and modernization plans for evolution of information systems. The Enterprise Strategy is what the Army must do to "Win the Battlefield Information War." It is founded on ten principles: Focus on the warfighter, ensure joint interoperability, capitalize on space-based operations, digitize the

battlefield, modernize power projection platforms, optimize the information technology environment, implement multilevel security, acquire integrated systems using commercial technology, ensure spectrum supremacy, and exploit modeling and simulation. The desired capabilities for 21st century systems are open systems with full global connectivity to national systems, situational awareness and a common picture of the battlefield, integrated voice/data/video/imagery, integrated Position/Navigation (POS/NAV), multilevel security, automated tracking of equipment and personnel assets, immediate access to sustaining base support systems, multiband/multimode radios, battlefield agility, strategic deployability, and artificial intelligence.

Inadequacy of Threat Assessment

Most of the documents describing the above-mentioned Army systems do no more than mention enemy threats; there is no assessment of threats such as jamming. The theme seems to be that by acquiring these new information and communications capabilities, we will prevent the enemy from having battlefield superiority. However, the very same documents claim that most military system products of the future will be based on or will use commercial off-the-shelf (COTS) products, which means that an enemy can acquire this technology as well.

Insights from Operation Desert Storm

Though the ground battle in Operation Desert Storm (ODS) was short, it offers insight as to the future importance of near-real-time data throughout the battlefield. It also begins to suggest how distributed data management technology might act as a force multiplier in future conflicts.

Most units that deployed during Operation Desert Storm did not have data distribution capabilities, and so the majority of communications were by voice over Combat Net Radio (CNR). When data distribution was used, its role was mainly to transport electronic mail over the Maneuver Control System (MCS) of the Army Tactical Command and Control System (ATCCS). Currently, the Army exchanges data using long, complex, United States Message Text Format (USMTF) messages. To send a few bytes of information often requires a very long USMTF. It was estimated that it would take approximately 60 USMTFs to populate an ATCCS map display that could be provided by one efficiently coded message.

Without a data distribution system, the Army command posts continue to use grease pencils on acetate map overlays to track friendly and enemy forces. This information, transcribed manually from radio and written reports, is (1) prone to errors, (2) often out of date, and (3) not synchronized with other command post maps. Current technology could provide graphical map presentation of situational awareness information including maps needed at different command levels at different resolution with the same information appropriately aggregated and presented.

ODS was the first war in which inexpensive, accurate, and timely position navigation information was available at the combat vehicle level; Global Positioning System (GPS) receivers made this possible. It was this key capability that allowed U.S. forces to move so confidently across the desert, particularly at night. Although the command and control systems at battalion and below did not have a data distribution system, critical information (GPS position and description of enemy forces being engaged) was communicated by voice using CNR. Both ODS and related exercises have demonstrated that platoon leaders and company commanders, having no staff, must either transmit critical information to the battalion level for aggregation and reporting to higher commands during pauses in the battle or must stop directing the battle in order to send the information [ASB, 1992]. Thus, the transmission of information at a critical time may be delayed because people cannot conduct a battle and send information at the same time. Several Army organizations, including the ADO, are addressing this issue. The solution includes automatically collecting GPS information, target laser information, and the information from automated sensors (for sensing the health of a fighting vehicle, resource usage such as munitions and fuel expended, etc.) and distributing the information to the relevant organizations both horizontally and vertically to contribute to situational awareness, near-real-time intelligence and targeting, and logistics.

Other insights from ODS include the realization that the current Army practice of sending status information in the form of situation reports (SITREPs) up the command chain, a step at a time, results in the corps commander receiving SITREPs 8 hours old from the forward line of battle; although critical information is often transmitted by CNR. With the increased use of indirect fire weapons, joint and coalition forces, and a nonlinear battlefield geometry there is a recognized need to send force location information everywhere in a timely manner for force synchronization and to protect against fratricide. It was also observed that overhead intelligence information is sent down the command chain and often does not arrive in time to be of use to the front forces, as was the case in the 73 Easting battle.

Much of the "fog of war" exists because commanders have different views of how the war is progressing due to differences in their available information. Part of the problem is that information is not "pushed" through the C2 system and made equally available at different command levels, and "pulling" or requesting the information when it is needed has not been satisfactory because of the communication lag; thus, such information may arrive too late to be of use in decisionmaking. The current limited communication bandwidth would be exacerbated by a data distribution system based on USMTF messages that are wasteful of the bandwidth. With limited bandwidth, there have been many concerns about how to maximize the use of the communications resource by determining what is the most important information to push. Studies such as the 1985 study of the Division Commander's Critical Information Requirements (CCIR) have been performed to attempt to optimize the use of the communications channel [USA CCIR, 1985].

Fears have also been expressed that sending too much information (e.g., through replication of databases) will cause an information overload and overwhelm the users. These concerns confuse the issue. Appropriate information and knowledge management techniques such as abstraction, aggregation, and filters can provide users with information in accord with their needs as well as be responsive to changes in their needs. Indeed, these techniques are needed regardless of whether information is local or remote; the difference is that if the information is local, it can be acquired and accessed in a more timely fashion.

The successful use of advanced technology in ODS is an indicator that the tempo of the battlefield will continue to increase with technological advances as military and commercial products become less costly and more available to everyone. Expected advances include faster, more accurate weapon systems and increased use of indirect fires, increased communication connectivity for command and control systems, increased use of GPS to accurately determine battlefield positions, higher imagery resolution, faster image interpretation, and rapid fusion and dissemination of intelligence information.

Focus of Study

The focus of this study was to examine data asynchrony assuming full replication of dynamic command data from battalion to corps. Partially replicated data were assumed for units below battalion and the study does not address how to achieve consistency, timeliness, and robustness when data are only partially replicated. The study does not address communication technology and media, or computer and security technologies, except for noting the context in which topics may

become issues. For simplicity, it has treated communications between units as point-to-point communications, though other alternatives are possible and are discussed below in Table 1.

Table 1 is a conceptual framework for tactical data communications. It shows three types of communications that can be used to send data messages: Broadcast, which is available to all who are tuned in; multicast, which is sent from one sender to a set of recipients through group membership; and point-to-point ,which is sent from one sender to one receiver. Each type may use encryption to ensure that messages are only "understood" by receivers that can decrypt them.

Broadcast communications are similar to direct broadcast television, require high bandwidth, reach a large number of users, can essentially be read all over the battlefield (from echelons above corps to the soldier in the trenches), and generally require no acknowledgment of message receipt. Multicast communications generally require medium bandwidth, reach a medium or limited number of users who share specific characteristics (e.g., logistics officers from brigade through corps), and usually require acknowledgment of the message. Point-to-point communications generally are designed to use low bandwidth, reach a single user, are generally used on the battlefield at battalion and below, and usually require acknowledgment.

Current efforts in Force XXI, ABCS, and digitization of the battlefield discuss two different ways to manage and use information on the battlefield. One way is nonhierarchical in nature and is still very conceptual. It is what theorists are calling "information carousels" that are organized according to functional areas such as intelligence, logistics, engineering, combat service support, etc., as shown

Table 1

Tactical Data Communications Conceptual Framework

Type of Communication	Type of Data	Distribution	Unit Level	No. of Recipients	Bandwidth Requirements
Broadcast	Information carousels (log, intelligence)	Horizontal	Entire theater	Large	Large
Multicast	Dynamic command data (pos, ORDs)	Horizontal and hierarchical	Brigade to corps	Medium	Medium
Point-to-point	Dynamic command data (pos, ORDS)	Hierarchical (horizontal in future)	Batallion and below`	Small	Small

in Figure 1. The "information carousel" data could be made available through broadcast to all units with proper access codes; e.g., units could tune directly to the intelligence or logistics station for real-time access to information without having to go up the command hierarchy for it. The concept is that the receivers tune in to get the data they need when they need it, and ignore the rest. It might also be possible for users to "pull" data from the carousels or from data centers that handle particular kinds of data by making direct data requests, or have data "pushed" to them on the basis of a profile they have registered with the data center.

Whether data are pushed or pulled on the battlefield, profiles, filters, or data aggregators must be used to prevent the end user from being inundated with currently unneeded data and to organize them according to a user's needs. Whether the data are pushed or pulled might be transparent to the end user. An advantage of pushing data everywhere is that replication can occur at less-intensive times on the battlefield when there is less contention for the communication bandwidth. The data would then be locally available when needed by the user in spite of jamming, communication delays, or outages.

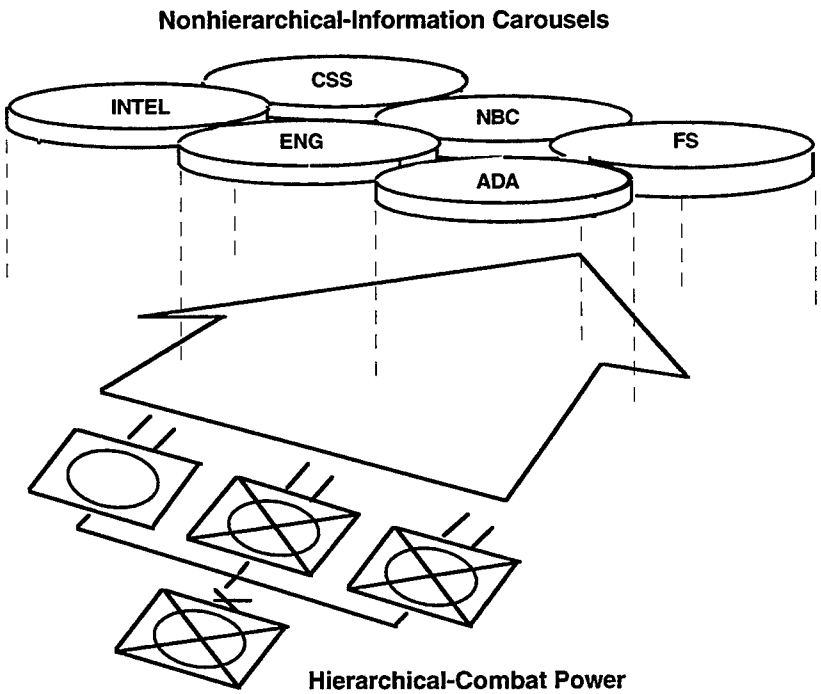

Figure 1—Concept of Nonhierarchical Information Carousels in Force XXI

The other Army usage of data is more hierarchical in nature. Dynamic data are created for the direct application of combat power on the battlefield and traditionally have been collected, managed, and used according to the unit hierarchy. The strict hierarchy is being extended by Force XXI to include horizontal distribution of unit data to other relevant organizations so that warfighters will have near-real-time common pictures of the battlefield for situational awareness. This will enable them to keep up with the increased tempo of the battlefield, get inside the enemy's decision cycle, and reduce fratricide. This study is focused on the replication of dynamic combat data but we give an example of a notional way of handling intelligence data and suggest that additional study be made of the carousel approach.

Questions remain as to whether total data replication is a good idea at the lower echelons of the battlefield (e.g, battalion and below), due to the security risk of being overrun by the enemy or of limited bandwidth, or whether there is even a real need for a lower-echelon unit to have situational awareness of distant parts of a nonlinear battlefield. Army doctrine has also traditionally called for each command level to look down only two levels. On the other hand, the type of military operations the Army is considering could have units moving under very flexible command structures in varying depths in the same region. Restricting data flow by a hierarchical scheme reduces a user's ability to have a "common view" of the battlefield in the sense of information—not just terrain graphics and deployment. Having wider access to data/information may reveal opportunities that are not obvious to commanders at different levels of the hierarchy or in different geographic locations.

If data are only partially replicated then there is a need for new, robust algorithms that will determine message interchange groups based on doctrine, mission, battle plan and orders, battlefield geometry, etc. These communication groups would be constantly changing as the units changed positions, plans, and missions.

Technology Areas

Management of Distributed, Replicated Data

Commanders at all levels require a timely common view of the battlefield to improve their capability to assess, plan, and execute the battle. This need for both horizontal and vertical data distribution and access is called for in all of the future-looking endeavors mentioned above. The concept is for warfighters at all levels to be seamlessly connected within their Service and jointly, across all

functional areas including Combat Support and Combat Service Support areas. This concept is also extended to efforts involving coalition forces.

Distributing replicated data appears to be a straightforward way to accomplish this need, but deciding why and where to maintain replicated data and how to ensure their consistency are complex issues that need to be addressed. Current commercial Distributed Database Management Systems (DDBMSs) maintain consistency through the use of synchronization techniques at the expense of the timely availability of data. Use of these commercial systems is not acceptable on the battlefield because of the time delay imposed by synchronization even in a normal, peacetime situation. These delays will be further exacerbated in wartime due to communication data-rate constraints and hostile actions such as jamming and node destruction.

Database Consistency

From a technical point of view, the consistency of a single database managed by a nondistributed DBMS is assured through a concurrency control mechanism (usually using time stamping or locking techniques) that enforces serializability of concurrent transactions. This means that the read/write activities of concurrent transactions may be interleaved only if the resulting database states are the same as would have resulted if each transaction had been operating one at a time in the order in which they occurred. The concurrency control problem is different in a distributed replicated database because of the need for consistency of multiple copies of the database. The condition requiring that all the values of multiple (replicated) copies of every data item converge to the same value is called mutual consistency. Transaction schedules that can maintain mutual consistency are called one copy serializable [Bernstein and Goodman, 1985]. If a distributed DBMS supports one-copy serializability, then a user can access the same data from any copy.

We believe that one copy serializability is not a viable option for C2 distributed databases since timeliness is essential. Because many other distributed database applications cannot afford the delay of one copy serializability, many current research activities are addressing alternatives, particularly solutions based on domain or application semantics. Some of this work is discussed in Section 3.

A major characteristic of the methodology we describe below is how it changes the data consistency problem from that of globally maintaining consistency among replicated databases at geographically distributed command and control sites into a local consistency problem at each site. The result is to increase the timeliness and availability of replicated data at the local site, but at the cost of

increased complexity in ensuring consistency at each local site. However, we believe that the data asynchrony problems caused by distributed data replication and communications delay due to jamming are the same as problems that must be addressed at the local site while interpreting and using data that contain errors, inaccuracies, or misinterpretations, and may differ among multiple sources. Indeed, even without data replication, there remains a data asynchrony problem in acquiring data from heterogeneous data sources because there is usually no guarantee that the data acquired were all previously updated at the same time or within the same time frame.

Convergence Toward Consistency

Our goal is to make replicated data available at distributed C2 sites in near-real-time or as quickly as possible, at the expense of consistency—some of the "replicated" data at any time may not be consistent in the sense of one-copy serializability. However, we want to insure that the replicated databases will converge toward consistency with time rather than diverge. This is a difficult problem that cannot be easily solved (e.g., by furnishing a previously jammed C2 node with the latest copy of the database from an accredited server) because many decisions and actions may have been made locally by the C2 node when the data were "inconsistent." It will be necessary to keep track of these dependencies and to propagate appropriate adjustments or corrections.

A side benefit of replicating data and the software to manipulate them will be robustness. A C2 commander at a node that is down or disabled will be able to carry on operations by connecting to a remote alternative node. Also, an isolated node or partitioned network may be operable with gradual degradation over a certain period of time (given no drastic external events). Finally, being able to operate robustly with calculated degradation could enable U.S. forces to employ jamming in a smart way—long enough to disrupt the enemy operations but not long enough to seriously degrade our own operations.

Related Efforts

In performing this study, we have focused our efforts on the "situational awareness" needs of Army tactical commanders from platoon through corps. "Situational awareness" data include mission plan and progress, represented as machine-processable operational orders (OPORDs), enemy situation, friendly situation and status, and terrain and environment data.

When we began this research project for ARPA several years ago, our literature search revealed no relevant distributed data management efforts addressing these issues. We did find efforts that used primary copies and secondary copies of replicated data where the primary copy was the most recent copy. We also found approaches that provided future times or dates for synchronization of data changes (e.g., all changes collected during the preceding day become viable at 8:00 the next morning, or these data go into effect on X hour on day Y). None of these approaches were relevant to the C2 problem. Other efforts (e.g., in the Computer-Aided Design/Computer-Aided Manufacturing (CAD/CAM) area) were directed toward domain-specific rather than general solutions. We have taken this approach for the C2 problem solution.

Over the past year we learned of a related research effort being carried out at the U.S. Army Research Laboratory (ARL) by Dr. Sam Chamberlain and associates [Chamberlain, 1990, 1994]. Their Information Distribution Technology (IDT) research program is addressing information exchange between fighting-level forces (i.e., brigade and below) constrained by relatively low-frequency tactical radios that do not support the high bandwidths common in most modern computer networks. They look on this as a unique environment in which computer processing power grossly outperforms communications power. The IDT experimental prototype implements a model-based command paradigm that uses a three-pronged approach to drastically reduce the average bandwidth use at the expense of computing: (1) Information is stored and exchanged in a common and terse form based on data standards and abstractions of military concepts; (2) a set of Information Distribution Commands are used to control the flow of information within and between tactical nodes so that only significant information is exchanged via the limited communications resources; and (3) a connectionless transport layer communications protocol, named the Fact Exchange Protocol, exploits the broadcast nature of combat net radio and provides efficient communications of database transactions. Instead of sending updates periodically (as is now done), they advocate the use of triggers and commander's criteria for sensing when changes to the data model are significant enough to be sent to others. This type of approach is also used by the Distributed Interactive Simulation (DIS) community to reduce the number of messages sent, by having simulations dead-reckon their objects' movements and comparing the results to simulation movement plans to determine if the variance is significant enough to require a position change message.

2. Approach to Replicated Databases for Robust Command and Control

Section 2 is divided into three subsections: The first discusses design concepts beginning with a design overview and principles; the second presents four types of events that are essential to situation awareness and describes how these four events would be handled; and the third summarizes the issues raised by these example events.

Design Concepts

The objective of the design and approach described below is to furnish commanders at all command levels with a timely, much more complete and consistent picture of the battlefield than they currently have. The design is intended to make them aware of possible inconsistencies when they occur and to help them keep track of actions that were influenced by possibly inconsistent data.

Design Overview

The design is based, in general, on organizing the battlefield database into sections to be managed in accord with the command and control unit structure on the battlefield as shown in Figure 2. Each C2 unit will have a software manager responsible for its software methods and the part of the database that it owns. Each manager will have responsibility for replicating its relevant data changes quickly at other C2 nodes throughout the system.

The synchronization mechanism between C2 nodes is weak, which allows decisions to be made and actions taken in near-real-time based on inconsistent data and inferences. It requires management of historical data, keeping track of actions that are based on the use of possibly inconsistent estimated data, and methods for propagating changes when consistency is later achieved. The synchronization method uses sequence numbers issued by each manager to enable the manager to know which data updates have reached which nodes.

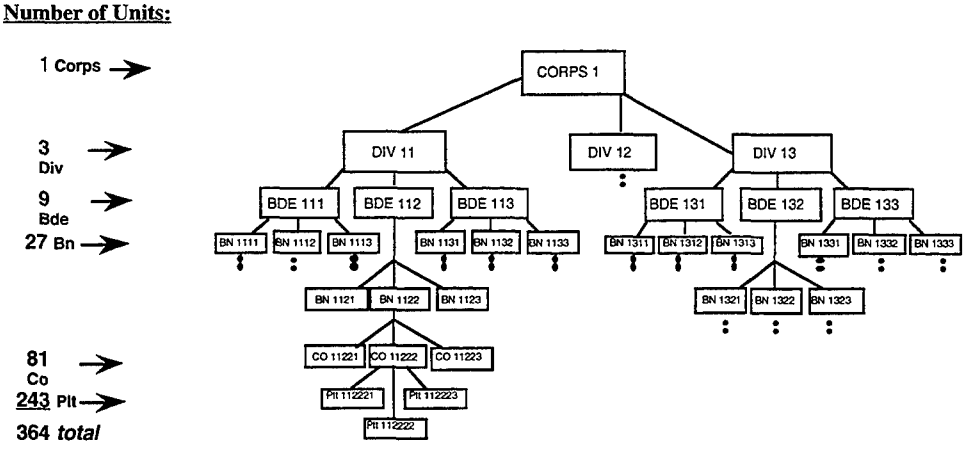

Figure 2—Corps Through Platoon Command Hierarchy

The overall result is support for timely data that at any point in time may be inconsistent but will converge in the longer timeframe toward a consistency or degree of correctness with respect to the real world. This means that historical real-world events can be reconstructed in the sequence in which they occurred at distributed sites, including actions that were taken as a result of inconsistent data. The design manages asynchronous information in such a way that it is available for decisionmaking and eventually converges toward real-world consistency. Convergence is ensured by using knowledge-management techniques to propagate new data to "correct" the older inconsistent data that were linked to and may have affected previous actions.

Command and Control Data-Handling Principles

Our approach requires eight general data-handling principles, which we describe below. These principles are compatible with the Defense Information Systems Agency/Joint Interoperability Engineering Organization/Center for Software (DISA/JIEO/CFSW) directives for DoD data standards and data.

1. Each data value and its metadata are "owned" by some responsible organization.
2. Each data value is entered into the battlefield system only once by its owner but may be replicated many times.
3. Each data value must have, associated with it, metadata that, at a minimum, include a unique data identifier, a data owner identifier, time of creation, security classification, and confidence level. (These metadata may be maintained for the entity (row or record) rather than by field value within

the record.) Derived data (data calculated from other data) must have metadata that describe their data sources and derivation method or algorithm. This will yield a derivation audit trail.

4. Each data value must be an allowed value of a domain of a data element or field that is associated with a DoD standard data element (SDE), where the SDE is an attribute of a DoD-recognized data entity discussed below.
5. Data are never destroyed by overwriting; they are always appended as new data.
6. Each message has a unique sequential identification number.
7. The normal mode of message acknowledgment is by liveness reports and negative acknowledgment (i.e., a site will request data it has inferred it is missing).
8. Updates to position location, plan changes, resource levels, etc., are normally provided by exception. Changes are compared to expectations within a variance defined by criteria and only propagated when they exceed thresholds.

Historical Data Management

This approach requires methods (software procedures associated with data objects) to maintain persistent historical data. Data are not destroyed by overwriting but rather new data are appended. Data may be archived when they are old. Each piece of data is entered into the system once, associated with a source organization, time/date, unique identifier, and receivers (those to whom the data were sent). Missing data can be inferred or estimated on the basis of historical data and thereby become "hypothetical" data to the system. Decision support tools can take advantage of historical data, making use of trends, replays, and analyses of past performance to get tactical insights, using history to review current courses of action (COA) and to explore others.

Knowledge Managers and Mediators for Army Unit

In Section 1, we discussed two ways the Army sees information on the battlefield being managed and provided (see Figure 1). One is through nonhierarchical information carousels (organized according to functional areas such as logistics or intelligence) and the other through hierarchical and horizontal distribution of dynamic combat data. This study is focused on the use of data replication to manage and provide dynamic combat data throughout the battlefield.

Our design supporting the replication of dynamic combat data requires that a Knowledge Manager for a C2 combat unit (KMC) exist for each C2 unit (e.g., battalion, brigade, division, and corps). The KMC manages all data unique to that unit through the Army Common Operating Environment (COE) software augmented by any unit/mission-specific software, including decision-aiding software. The KMC can be thought of as an automated assistant to the unit commander/staff, etc., for carrying out the unit's mission. Changes to the unit's database are initiated only through the unit's KMC. In addition to managing the unit's data, the KMC is responsible for managing the propagation of the unit's data to every other C2 unit. The replications are handled through Mediator(s) for the C2 unit (MDCs) at every other C2 unit. The KMC and MDCs for a given C2 unit must have identical software. Any changes to a unit's KMC management software needs to be synchronized with software changes to its MDCs at every other C2 unit.

For discussion throughout this report, we consider a C2 unit's data to be managed logically by a KMC at a single node where a node is a configuration of software/hardware accessible through a Defense Information System Network (DISN) address. In reality, a unit could have multiple physical nodes and the physical responsibility for the data could belong to multiple managers who are coordinated by the C2 unit's KMC. For this discussion, think of each C2 unit's commander/staff as using its KMC on one physical node.

At each C2 node, there will be a number of MDCs to manage the replicated resources of each remote C2 node. An MDC will make its remote node's data available to the local node commander and staff through the local node's KMC. An MDC can use its methods to infer or estimate missing data, changes in plans, etc., when requested to do so by its local node KMC. An MDC serves as a bridge between the local node's KMC and the KMC of the corresponding remote node. An MDC knows (1) how to receive and handle the replicated data and methods from its remote KMC, (2) how to apply the same methods (as would be used by its remote KMC) to those data, and (3) how to interface with its local node KMC to provide data or best guess estimates when information is missing.

Figure 3 shows an example distribution of KMCs and MDCs in a subsetted view of a replicated C2 database comprising an intelligence node and a corps node commanding two divisions, each having two brigades. In Figure 3, there is a KMC for Brigade 111 at the Brigade 111 node and MDCs for Brigade 111 at every other node. Each MDC receives data and information from its remote KMC and receives requests and exchanges information with its local KMC and other local MDCs. The MDC can incorporate only new data received from its remote KMC

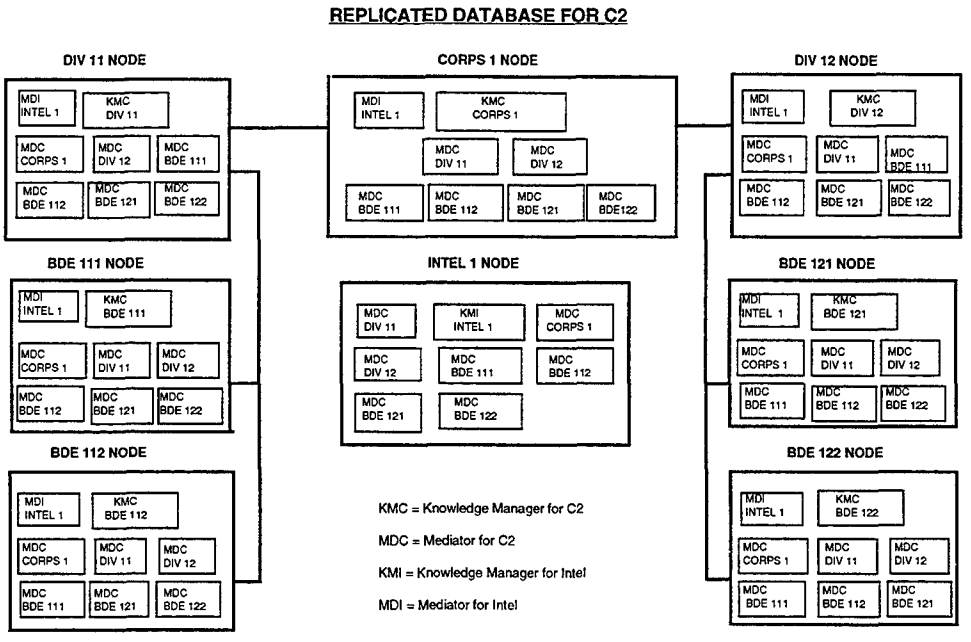

Figure 3—Example Distribution of Knowledge Managers and Mediators for Replicated C2 Databases

or data it has inferred or estimated at the request of the local KMC (if, for example, the MDC data are old because its KMC has been unavailable due to jamming). Data derived by inference or estimation are marked as such.

A KMC sends a data message to other KMCs by sending the "replicated" message to its corresponding remote MDCs and instructing its MDCs (on nodes whose KMCs are designated as receivers) to deliver the message to their KMC.

The design also includes an intelligence node—with a Knowledge Manager for Intelligence (KMI) node and a Mediator for the KMI at each C2 node (called Mediator for Intelligence (MDI)).

Special handling of intelligence data is necessary because intelligence data can be sparse, ambiguous, at variable resolution levels, maliciously misleading, of varying security levels, etc. There are conflicting needs to have intelligence data available everywhere as soon as possible but also conflicting requirements to have intelligence data centrally managed under integrated control where they can receive expert interpretation and analysis. Control is needed to specially downgrade highly classified data for release at lower security levels appropriate to different KMCs. It may be that, for security reasons, not all intelligence data can be replicated at every node, even within a multilevel secure DBMS, because

the forward areas may be too susceptible to enemy capture. Though we note this security risk, our design supports full replication of intelligence data.

Design Rationale for Data Distribution to C2 Units

Several points about the KMCs and MDCs (and KMIs and MDIs) introduced in this study need further clarification.

The ARPA sponsor requested a study in which all data would be pushed everywhere in near-real-time to increase their availability, based on anticipated technology advances in computers and communications. We extended this study requirement to include replication of the C2 unit software required to manipulate the data. This is not so different from the direction in which the Services are currently heading through compliance with the DISA Technical Architecture Framework for Information Management/Technical Reference Model (TAFIM/TRM). The TRM includes the concept of layers of shared/reusable software at the application level, application support level, domain level, system support level, etc. A Joint COE has been agreed to by the Services and will be the core software for the J6 GCCS. The Army Program Executive Office (PEO) for Command and Control Systems (PEO/CCS) is developing an Army COE based on the COE that will be the core software for all Army ABCS systems. The concept is that if all systems are compliant with the TAFIM/TRM and the TAFIM Human Computer Interface (HCI) guidelines, and these systems incorporate the layered COE, then it should be possible for C2 applications to operate on any C2 node. This is the approach we are supporting in this study. From battalion to corps, all C2 software to serve all command levels should be available on all platforms along with the relevant data.

This study assumes that every C2 node has replicated data and resources on every other C2 node. In an example Army hierarchical organization structure—1 corps having 3 divisions, each division having 3 brigades, each brigade having 3 battalions, each battalion having 3 companies, each company having 3 platoons—there are a total of 364 units.

Since the company and platoon are at the battlefront fighting, have no dedicated staff, and for security reasons may not have access to total information, our initial design addresses total data replication from battalion through corps. We will address company- and platoon-level data distribution in more detail in a future effort. The replicated data distribution for them will need to be algorithmically based on doctrine, mission, battle plans/orders, battlefield geometry, etc. It may also turn out that battalion data distribution needs are more akin to those of the company and platoon than to those of the brigade, division, and corps. On the

other hand, there is no technical reason why full data replication could not be extended downward to platoon level.

The current study does assume that platoons and companies have some kind of automated data distribution system (i.e., an integrated GPS, communication, and computer device) that can make decisions or be used by the platoon leader or company commander to make decisions about when a position location change, an enemy sighting, a SITREP, or a synchronization message should be reported up the chain (platoon to company, company to battalion) and horizontally among platoons and companies. However, we are not addressing KMCs and MDCs at lateral platoon and company levels. We will, however, be discussing the distribution of messages up and down the command chain to companies and platoons.

Robustness

The use of historical data and total data replication makes for a very robust C2 system. Essentially a commander and his staff would operate through their KMC at the command post (CP) node they select. They could create an MDC, at each of their other CP nodes, to act as a surrogate KMC. By using replication, a commander in trouble on one CP node may choose to command from another CP node. For example, when a CP needs to move, the KMC could be "moved" by having the commander designate a surrogate KMC at another of his CP nodes to be the active KMC. When the moving CP node stops and comes back on the air, it would become a surrogate KMC and request all the information it missed during the move. After it is brought up to date, the commander may wish to reestablish it as the active KMC and the other CP node as the surrogate. The commander could also command through his respective MDC on another unit's node.

C2 doctrine could require each commander to designate one or more surrogates as backups. In addition, the entire database could be backed up at a sanctuary, e.g., on a satellite, in the rear, or in the Continental United States (CONUS).

Transaction Management

When a new message is received, it is recorded in a log of "received messages" maintained by the KMC. When the message is ready to be processed, a transaction is started (a transaction can be thought of as a program as discussed in the appendix). Whenever data processing is done by the KMC (or KMI), either automatically when it receives a message or interactively when supporting a

commander and/or his staff, a transaction is started. A log of all new data created and entered into the KMC database is also maintained. After the transaction has been known to correctly complete, the last operation is to replicate these data by sending them to all of the KMC's MDCs.

The major steps in the transaction processing are:

1. Begin transaction,
2. Open log for new data,
3. Enter transaction information into data log (include message identification if transaction is to process message),
4. Perform data operations,
5. Complete last data operation of the transaction ,
6. Close and save data log, and
7. Send log of new data to all MDCs.

After a transaction has completed, the user may want to send one or more data messages to other nodes, or the KMC may be automatically triggered to do so. These messages will contain data from the KMC database and/or local MDC databases—data that should have been previously replicated in databases at the remote nodes. The message will be sent to remote nodes by sending it to all the KMC's MDCs. The MDCs located on the nodes of the message recipients will "deliver" the message to the node KMC (a message recipient).

Note that when the message is sent, that information is recorded in a log of "sent messages" maintained by the KMC. Each of its MDCs will maintain a replication of this "sent messages" log by entering the relevant message information into its own log upon receipt of the message.

When the message is received by the MDC:

1. The MDC receives the message, and logs it into its "messages received" log and into its replication of the KMC's "messages sent" log. If the MDC's local node KMC is a recipient of the message, then the MDC delivers the message to the local KMC.
2. The recipient KMC will check the data identifiers in the message with its local MDCs to ensure that the data are present on the local site through previous replication messages. If the data are not present on the local site and are labeled hypothetical, then their node is probably unreachable and the data is entered in the relevant MDC database (along with their hypothetical

tag and information). If the data are not present on the local node and are not hypothetical, then it can be inferred that the relevant MDC has missed a data-replication message, the receiving KMC will request the missing message from the MDC's KMC, and the data will be entered as "without official confirmation" in the MDC's database.

3. Since the data in the message are uniquely identified, they will not be reentered into the MDC database(s). The message sequence number and date/time sent will be entered as metadata and linked to the data record(s) of the message data in the MDC database(s).
4. The message will be logged into the message log of the local KMC, including metadata identifying its sender, sequence number, date/time sent, date/time received, and date/time processed.

Synchronization Through Use of Sequence Numbers and Liveness Reports

KMCs assign unique consecutive numbers to their messages. This enables an MDC to determine when messages are missing and to request them from its remote KMC. Ways for the KMC to infer that messages may be missing include failure for a KMC to receive a "pushed" report (e.g., a SITREP) within an expected time interval or the receipt of information indicating that a node is down or that jamming has occurred in a particular area.

Liveness reports are used by the KMC to affirm the health of each of its MDCs as well as to support synchronization of the KMC's replicated data. Liveness reports are sent by MDCs to their KMC, at prearranged time intervals or on request through a liveness request message. In a liveness report, the MDC sends the highest message number it has received, the highest sequential message number it has received, and requests for any missing messages. The liveness report is used by the KMC (1) to confirm that another node is reachable, (2) to know the highest consecutive sequence number reached by all of its MDCs (which can be used to determine what data are safe to archive), (3) to resend missing messages, and (4) to bring MDCs up to date after their nodes have been unavailable. A KMC can make the same inferences that an out-of-touch MDC would make (on the basis of the confirmed replicated messages that the KMC knows the MDC has already received). Furthermore, this enables the KMC to send out an appropriate alert to other units if the inferred state would be very different from "reality."

Messages can be labeled with the kind of acknowledgment (ack) or negative acknowledgment (nack) that is expected. The message can request an ack within a certain time window (including immediate response) or no acknowledgment unless the KMC or MDC is missing a consecutive sequence number. If a sequence number is missing, a request for missing messages would be made (a nack). If an ack is expected within a certain time window and is not received during that interval, the message may be resent n times (n being determined by some criteria).

Because immediate acks consume limited communication bandwidth (especially during critical times when there are communication failures), nacks are preferred. However, during a crisis it may be essential to synchronize nodes to coordinate a plan. A message requiring immediate acknowledgment is a good way to do this.

When a KMC has determined that it is safe to archive the data, it archives its copy of the data and sends a data archive message to all of its MDCs.

The KMC must maintain a table containing entries for each of its MDCs recording the highest consecutive message number each MDC has received, the date/time the information was sent by the MDC, and the date/time the message was received at the KMC. This table must be sent as a replicated data message to all of the KMC's MDCs. Determining when to send the table will be based on shared criteria (e.g., periodic time interval, time lapse after requesting a synchronizing response, or whenever the table changes).

This information is especially important to surrogate or backup nodes for the KMC and, if there is one, to a sanctuary MDC. These are the MDCs that, by agreement, would be prepared to take over the KMC responsibilities should the KMC move, go down for maintenance, or fail. Since the KMC is a site designated by the commander, even if the network should be partitioned, the partition of MDCs that did not include the KMC would continue to operate as MDCs. However, they would not receive KMC messages, though they could infer or estimate what the KMC would do, especially if coexisting MDCs representing immediate subordinate units were receiving messages from their KMCs. If the commander wished to move his KMC, for example, when the command post was planning to move, the coordination between the KMC-to-be and the current KMC would be done by the KMC. The current KMC would send to both the KMC-to-be and the network name server a message containing the future time at which this changeover would occur.

Although we have addressed in general how the replicated distributed data management message system would work, we should not lose sight of the fact

that it is part of a larger system that includes voice communication as well. For disseminating critical information or synchronizing critical activities, voice communication will probably still be the preferred medium. However, to enable the data distribution system to make the best estimations and inferences, it is always desirable to enter the data exchanged by voice into the system.

Maintaining Consistency at the Local Node

The design presented for using replicated databases to ensure a robust C2 system changes the data consistency problem from that of maintaining consistency among replicated databases at geographically distributed C2 sites into a local consistency problem at each site. Our approach allows the commander and his staff to use the near-real-time data immediately in decisionmaking and execution, because it gives them support for tracking the data dependencies in decisions and propagating critical data changes to the relevant actors. If data are late or missing, they may be estimated or inferred by the relevant MDC, labeled as hypothetical, and used in the KMC decisionmaking process.

To enable convergence toward consistency, the system must (1) recognize when the data it generates are hypothetical (e.g., estimated or inferred), (2) record this awareness as part of the metadata, and (3) set up the proper triggers or alarms to be activated when data arrive in the future that could improve on hypothetical data that affected earlier actions. As stated above, it matters little whether the new data are arriving late due to jamming or node outages or are correcting earlier corrupt or badly estimated data. These conditions can all be handled in the same way.

Most importantly, some mechanism such as a "check list" is required to represent the hypothetical data in a way that allows them to be manipulated and understood by the Decision Support System (DSS) and transformed into a presentation that can be easily understood by the DSS user. Whenever important decisions are being made, DSS users could be asked to peruse the check list and indicate which of the hypothetical data entries have influenced their current decision or action. Furthermore, if decisionmakers desired, they could establish specific thresholds or rule changes that would trigger propagation of the information. Propagation could include (1) propagating the new data to an organization that had received the hypothetical data, (2) propagating data derived from the new data to replace data previously derived from the hypothetical data, or (3) attempting recovery from an action that resulted from a decision based on the hypothetical data (e.g., a change in a Fragment Order (FRAGO) or Operational Order (OPORD)).

The decisionmaker could be aided in this task by a smart transaction processor that turned on a check list indicator whenever it "touched" hypothetical data. The DSS would be supporting the decisionmaker in a way he has never experienced—helping him to keep track of the dependencies between the use of hypothetical data and actions.

A real concern with local transactions is to decide how transactions should be structured and supported, and when and how they can be interrupted to handle higher-priority transactions. Several conflicting needs are involved. One relates to the speed of data replication versus the amount of message traffic. It is desirable to deliver the new data to the KMC's remote MDCs as quickly as possible but it is also important to minimize message traffic. Another requirement is the need to complete decisions as rapidly as possible versus interrupting decisions when relevant data changes arrive in order to base decisions on the most current data at the expense of possibly having to back out a long transaction or at least extend its time. Another consideration is whether an entire decisionmaking session or all the actions resulting from an incoming message should be handled as a single long well-defined or ad hoc transaction (though it could be a composite or nested transaction) compared to an approach that would break up the actions into individual smaller transactions.

More attention needs to be paid to these issues, and they are discussed further in Section 3.

Types of Messages

In this study, we have addressed only point-to-point messages. This report does not consider broadcast or multicast messages, though the message requirement described in this report could be supported by a multicast protocol.

Assurance that messages have been received is carried out at the application level. As discussed above, messages requiring acknowledgments within a time constraint are supported but the more usual mode of operation is negative acknowledgment. An MDC will know if it is missing one or more messages (through sequence numbers) and will request missing messages from its KMC.

Since all the data in messages are based on DoD data standards, the messages can be self-describing. The format of a message's contents will be described using a single common syntax. Frequently used types of messages (e.g., position reports, situation reports, or OPLANS) may have a registered format.

One way to categorize types of messages is on the basis of the command and control functional areas (or more detailed sub-functional areas) such as fire support, intelligence, combat operations, air operations, maritime operations, combat service support, and general administrative. These are the major types of categories for USMTF messages.

Another categorization that is useful for understanding the design is on the basis of types of information exchange between the design entities (KMCs, KMI, MDCs, and MDIs) we have described above. This categorization is shown below.

All messages begin with a preamble containing the following information:

Date/time sent

Sender identification

Date/time received

Receiver/identification

Message sequence number

Message type: the types could include those enumerated below plus others such as "position location update," SITREP, OPLAN, email, self-describing

Message instruction code: "ack required within n time," "ack required by time N," "nack only required," etc.

Messages sent from a KMC to its corresponding remote MDCs (or from the KMI to its MDIs):

1. Replication message: sent after a transaction has completed.

   Message includes preamble, transaction number, new data.

2. Data update message sent to other KMC(s) (via MDC(s)).

   Message includes preamble, new data, or new data identifiers.

3. Data request message sent to another KMC (via a MDC).

   Message includes preamble, description of data requested.

4. Response to a data request message.

   Message includes preamble, preamble of message request, requested data.

5. Liveness (status) message.

   Message includes preamble.

6. Status table message.

   Message includes preamble, status table data.

7. Data archive message.

Messages sent from a MDC to its remote KMC (or from the MDI to its KMI):

1. Liveness report in answer to a liveness message.

   Message includes preamble, status information.

Basing C2 Data on DoD Data Model and DoD Data Standards

The DoD 8320 series of documents describes and promulgates DoD data standardization policies and procedures [DoD 8320.1, 1991]. The process ideally calls for the development of functional area process models (preferably using Integration Definition Language for Function Modeling (IDEF0) methodology [FIPS 183, 1993]) developed to the detailed level of specifying data interchange requirements. The data interchange requirement entities and attributes then become inputs to a data modeling effort (preferably using IDEF1X methodology, [FIPS 184, 1993]).

The IDEF1X data modeling methodology graphically presents the relationships between entities, and entities and their attributes, and supports a very simple "IS-A" relationship through a categorization concept. It is lacking in support for some advanced object-oriented concepts such as multiple inheritance, multiple roots, and polymorphism. It also lacks support for part-whole and other domain-specific relationships and the capture of business rule information in computer-processable form. All of these shortcomings are being addressed by a new Institute of Electrical and Electronic Engineers (IEEE)IDEF1X Working Group.

The data model for the functional area is then compared to the DoD Data Model for reuse of entities and attributes. Any entities and attributes not found in the DoD Data Model may be proposed as new entities and attributes in a proposal package that includes the functional area data model, identification of reused data element standards, and proposed standards for newly identified entities and attributes. The proposal package is reviewed by the Component supporting the data modeling and standards effort and submitted to the DISA/JIEO/CFSW process via the DoD Functional Data Administrator (FDAd) for the functional area that was modeled. The DoD data standards, which uniquely identify data elements through detailed description including domain constraints and usage,

are managed in a data dictionary, currently the Defense Data Repository System (DDRS).

The ultimate goal of the DoD data standardization process is to achieve interoperability and reuse of data throughout DoD (and with coalition forces), reduce data redundancy and inconsistency, and improve data quality and validity. Improving the data will improve the interoperability of the applications using the data.

Data standardization is an important part of the Army's goal to digitize the battlefield. The kind of seamless interoperability required by the Joint Staff C4I for the Warrior effort, and the Army and other Components' future plans requires data standardization. The Military Communications Electronics Board (MCEB) has concurred on accepting the C2 Core Data Model (which is a C2 view of the DoD Data Model) as a good starting point for data modeling of the C2, intelligence, and modeling and simulation functional areas. The C2 Core Data Model is a version of the C2 Common Hub Data Model developed by the North Atlantic Treaty Organization (NATO) with support from the Army ODISC4. [IDA, 1994]

If all of the C2 databases' and the "carousel" databases' (e.g., intelligence, logistics, and medical) schemas map their entities and attributes to the DoD standard entities and attributes, then data fields in messages can be self-defining. Message data fields would be composed of two parts—a standard entity/element identifier for the field and a data value (or encoded data value). The current IDEF1X and DoD standardization methodology and policy and procedures are based on the relational data model and the use of relational databases. In Section 3, we discuss issues having to do with the use of object-oriented database technology rather than relational database technology.

Data standards also include standards for data element domain values and symbology. For example, currently, without standards, the M1-A1 tank may be identified in different data collections as the M1-A1, the M1A1, the m1-a1, the m1a1, etc. When data element domain values are standardized, it is straightforward to encode them in a standard way to reduce bandwidth when transferring data over a network. Probably the only time encoded data will need to be decoded is on presentation to the user either in softcopy or hardcopy.

The ARL effort mentioned above, with a primary objective of constraining the use of communications bandwidth, developed some interesting ideas about how data could be encoded. In its IDT, information is stored within a factbase as a collection of many interconnected facts. "A fact is an instance of a pre-defined fact-type, or template, that is structured to describe an item, activity, or event

common to the battlefield environment. A fact-type consists of a header and one or more fact items. The fact items will be one of five possible data types: integers, floating point numbers, character strings, references (to other facts), and lists (a collection of any of the above data types)." As facts are entered into the factbase, each is assigned a universally unique identification number that is permanently attached to the fact and moves with it.

The fact identification (id) number serves as the surrogate key for each row (entry) of the relation. Thus, relations can conveniently refer to entities in other relations via fact ids.

The IDT factbase is a simple read access memory (RAM)-resident database that is a mixture of relational, network, and object-oriented database features. A fact id is a surrogate key to a record of facts (data). Lists of surrogate keys are used as pointers to other facts (similar to a network database) to greatly improve performance at the cost of redundancy and violation of relational normal form, although denormalization is tightly controlled. The output of a query to the database is not data fields but a list of the surrogate keys of the facts having data fields that match the query conditions. The surrogate keys are then used to retrieve the facts.

These concepts are consistent with the concepts and goals of data standardization.

Categories of Data

The ARL effort has also categorized a fact as being either a dynamic fact, a reference material fact, or a metafact. Each dynamic fact may be created or destroyed by the user, describes changing battlefield events and activities, and is associated with its factbase of origin (its host) by its fact id. Their reference material facts describe stable reference information such as is found in Tables of Organization and Equipment (TOEs) and have static fact ids common to all factbases. Reference material facts can never be created by the user (an example is an Army unit) and do not necessarily have only static information within the fact. Metafacts are facts about facts and are often used to represent hypothetical modifications of other facts (such as a hypothetical move of a unit to a new location as part of a "what if" exploration). In this example, a metafact would be created that referred to the existing unit fact but would contain an alternative location.

ARL researchers have developed a preliminary list of fact-types or primitive data abstractions and some relationships between them (as shown in Figure 4). They

stress the importance of the technologists working with the military scientists and operators to produce effective designs.

If we were to categorize the data concepts in our design, we might align the KMC and KMI data to dynamic data that may be associated with reference data with static fact IDs. An example might be a wheeled vehicle, which is a real-world object that cannot be created by the user and has many attributes that are static (e.g., dimensions, engine number) but some that are dynamic (e.g., location, fuel status, condition). We also might want to distinguish between types and instances, where an M1-A1 tank may be a type of tank and a particular M1-A1 tank an instance. In this case, the type information would tend to be static, whereas the instance information would share the static data and would also include dynamic data.

Our design uses much metadata, in fact more metadata than data. Except for the general definition of metadata as "data about data," there is no currently accepted taxonomy of types of metadata though this is an issue being addressed by a new IEEE Metadata Working Group. The International Organization for Standardization/International Electotechnical Commission (ISO/IEC) 11179

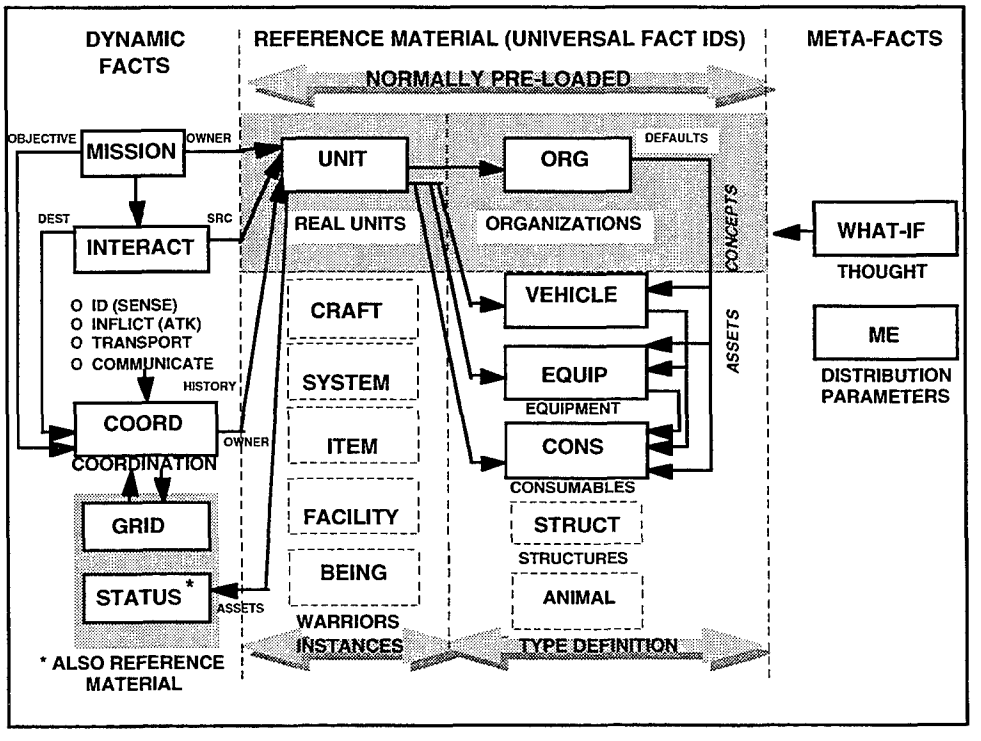

Figure 4—Fact Types and Some Relations (taken from ARL 1994 briefing)

document on Data Element Specification and Standardization has concurred on five kinds of basic attributes (metadata) of data elements. They are identifying, definitional, relational, representational, and administrative.

We need to consider whether the ability to categorize data provides some important advantage to the design before additional resources are devoted to addressing this issue.

Examples of Handling C2 Events That Support Situational Awareness

For purposes of illustration we will walk through, at a high conceptual level, four events necessary to support situational awareness including position and status of friendly units, position and status of enemy units, and mission (i.e., operation plan/orders). These events indicate the kinds of system activities required to pass information horizontally and vertically for near-real-time processing concurrently throughout the entire system. The examples will focus on highlights but will not enumerate detailed data that would be stored in a KMC database and passed to its MDCs.

The examples are:

1. A friendly platoon position location change exceeds its movement plan threshold. This event demonstrates how a position location change could ripple up through the hierarchy and at each command level would generate more aggregate data used as the basis for a new threshold decision.
2. While generating a situation report, it becomes necessary to estimate a missing or late report from a subordinate. This example illustrates how missing data can be inferred, how inference can affect actions, and how a dependency between action and inference is maintained for subsequently propagating consistency.
3. This example demonstrates a change in operation plan from top down and from bottom up to show propagation of critical messages for priority attention at all command levels or to selected levels.
4. In this example we show the propagation of an enemy position location report from platoon to corps, and the propagation of an intelligence report from above corps to platoon. This illustrates the roles of the KMI and MDIs.

Example One: Position Location Change of a Friendly Unit

In this first example, we describe in detail how a position location change of a tank can trigger changes up through corps. The "thresholds" mentioned for triggering movement reports to higher levels may be simple distance triggers (e.g., if the tank moved more than n meters over the past x time period, then alert), or more complex, such as a distance and direction change that suggests a change from the OPORD and possibly a need to change to a new FRAGO part of the OPORD or an indication that replanning may be necessary. The thresholds are flexible and are set by commanders at each level with respect to the OPORD being carried out. They are based on the CCIRs whereby the commander has explained the mission and his intent to his staff and wants to be alerted if conditions occur that may change either of these. These "conditions" are translated into "thresholds," which could be quite complex decision processes.

What we do not show, and have not addressed, is how individual tank and platoon position location information gets propagated to many different relevant areas to reduce fratricide. We noted that our corps example (Figure 2) contains 243 platoons. Assuming that half the platoons will have four tanks each, this configuration would yield a total of almost 500 tank positions that would need to be disseminated, for full replication to adjacent platoons and companies, Army fire control units, Army helicopters, close air support, other Components, joint units, possibly coalition units, etc. It may not be meaningful to disseminate individual tank positions, since the tanks may be continually changing position; rather, it may be preferable to disseminate the information at a higher level of aggregation, e.g., platoon or company. There is a tradeoff between the desire to disseminate what could be critical detail data to every level as fast as possible (which can increase the communication traffic and require sophisticated techniques to pick out the critical details from the chaff) and the need to delay dissemination until the information has been aggregated at a higher level, thus reducing the traffic and the processing. Also, the requirement needs to be examined closely. Reducing fratricide may require the identification of areas in which enemy and friendly forces are commingled at some resolution level and not the individual tank locations. However, planning a synchronized attack may require more than just the area; it may also require some indication as to direction of the tanks or platoon and their speed.

At the tank level:

— A tank has moved in a different direction than shown on its plan to avoid a terrain obstacle. Its new position is entered into the tank database. The movement is more than x meters in direction y, which exceeds a threshold causing a position change message to be sent to the platoon leader.

— A position change message is sent horizontally to other tanks, platoon leaders, etc., in the geographically relevant and mission relevant area.

At the platoon leader level:

— The message is received from the tank.

— The tank position change data is entered in the platoon database. A new center of mass for the platoon is calculated based on the message information and dead reckoning the locations of the other tanks in the platoon. The center of mass and direction are compared against the platoon threshold.

— The threshold is exceeded and the estimated locations of the other tanks are entered in the platoon database. A platoon position change message is sent to the company commander that contains all of the new data entries caused in the platoon database by the tank position change message.

— The same platoon position change message is sent horizontally to other platoons, companies, etc. in the geographically relevant and mission-relevant area.

At the company leader level:

— The message is received from the platoon.

— The platoon position change data are entered in the company database. Locations of other platoons in the company are dead reckoned and entered in the database as estimates with confidence levels.

— A new center of mass for the company is calculated based on the message information and dead reckoning the locations of the other platoons in the company. The center of mass and direction are compared against the company threshold.

— The threshold is exceeded. The estimated positions and confidence for the other platoons are entered in the company database. A company position change message is sent to the battalion that includes all the database additions caused by the position change message.

— The same company position change message is sent horizontally to other platoons, companies, battalions, etc., in the geographically relevant and mission-relevant area.

At the battalion-level KMC:

— The message is received from the company.

— The company position change and platoon information are entered in the battalion database. Locations of other companies in the battalion are dead reckoned as estimates with confidence levels. A new center of mass for the battalion is calculated based on the message information and dead reckoned locations for the other companies in the battalion.

— The center of mass and direction are compared against the battalion threshold.

— The threshold is exceeded. The new position for the battalion is entered in the battalion database and a message is sent to the battalion KMC's MDCs containing:

  — All the new data added into the battalion database as a result of the company message;

  — Instructions to the MDC (located on the node whose KMC is the commanding brigade of this battalion) to deliver the message to its local KMC.

(If the thresholds were not exceeded, then a message would be sent to the battalion KMC's MDCs containing the new data added into the battalion database as a result of the company position change message.)

At a battalion-level MDC:

— The message is received from the battalion KMC and the MDC database is updated to contain the message data.

— If the MDC's local KMC is the brigade command for its battalion, then the message is forwarded to the local KMC.

At the brigade-, division-, corps-level KMC:

— The message is received from the subordinate MDC.

— Locations of other subordinate units are dead reckoned through their MDCs and a new center of mass for the KMC unit is calculated based on the message information and dead reckoned locations for the other subordinate

units. The center of mass and direction are compared against the unit threshold.

— If the threshold is exceeded, then the new estimated unit position is entered in the KMC unit's database and a message is sent to the KMC's MDCs containing:
  — The new data added into the KMC database as a result of the movement message.
  — If the KMC is a brigade or division KMC, then instructions are sent to the MDC (located on the node whose KMC is the MDC's commander) to forward this message to its local KMC.
— If the KMC is a corps KMC, then the message is sent to its commander (e.g., Army Headquarters (HQ), joint task force, or coalition force leader).
— If the threshold is not exceeded, then a message is sent to the KMC's MDCs containing the new data added into the KMC database as a result of the message.

At a brigade-, division-, corps-level MDC:

— The message is received from the KMC and the MDC database is updated with the message data.
— If the MDC's local KMC is the commander of the MDC's unit, then the message is forwarded to the local KMC.

Example Two: A Situation Report

SITREPs flow from lower to higher command levels. They are required periodically, usually according to a schedule that assures that SITREPs from units at one level are expected at the next higher level before the beginning of the situation assessment and replanning cycle at the higher level. A SITREP consists of two types of information: summary status data about personnel, arms, logistics, training, intelligence, etc., and overall assessment information based on the commander's evaluation of the situation, his interpretation and plans for handling shortfalls and problems, and his evaluation of his unit's ability to fulfill its mission. The summary information can, in most cases, be automatically generated from the detail data, but the overall assessment is judgmental. When a SITREP is late or missing, it is this judgmental portion that is difficult to estimate and record because it requires human judgment using information that may not be in any database, such as how a commander plans to attack a problem, whom

he has spoken to about solving it, and what he expects in the way of support and from whom.

The personnel summary information includes authorized personnel by number and skill, assigned personnel by number and skill, those present for duty by number and skill, those in the hospital, those killed or missing in action, and the training status and schedule for different skill levels. The logistics summary includes information, based on the mission plans, about on-hand supplies and shortfalls for supporting the soldier (e.g., food, water, and clothing); munitions; fuel; maintenance facilities and parts (particularly for major weapon systems such as tanks); protective clothing and aids against cold weather and nuclear, biological and chemical attack; and transportation assets. The arms summary includes positions of units, readiness rates for major equipment and weapons systems in terms of those operationally ready/on-hand, and those operationally ready/authorized. The intelligence summary represents the enemy threat as understood by the unit's intelligence officer and commander. All of this summary information should be computable from information available in the subordinate level SITREPs.

The judgmental section of the SITREP addresses, for any low readiness rating, what is being done to remedy the situation, any outstanding problems that affect the mission that cannot be solved at this command level, and most important, the commander's overall assessment of the readiness and state of his unit in terms of the threat and mission.

Below we describe the propagation of SITREPs up the command hierarchy. This includes showing how information from missing or late SITREPs can be estimated by a KMC from the various MDCs resident on the same node, and how the later arrival of a SITREP, after actions have been taken based on the estimated information, will propagate changes.

At the platoon level:

— The platoon leader prepares the platoon SITREP based on and including the summary status of his individual weapon systems (e.g., tanks), enters it into his platoon database, and sends it to his company commander.

— The SITREP is sent horizontally to other platoons, companies, battalions, brigades, etc., in the geographically relevant and mission-relevant area.

At the company level:

- The company commander prepares the company SITREP based on and including the status of his platoons from his platoon SITREPs or estimated SITREPs, enters his new SITREP into the company database, and sends his SITREP to his battalion commander.
- The SITREP is sent horizontally to other companies, platoons, battalions, etc., in the geographically relevant and mission-relevant area.

At the battalion-, brigade-, division-, and corps-level KMCs:

- If all the direct subordinate SITREPs are available, then: The commander prepares his SITREP based on his direct subordinate SITREPs and his judgment; the commander subsequently stores his SITREP in his database.
- If the commander is missing one or more SITREPs from a direct subordinate(s), he then:
  - Estimates the information for that unit in his SITREP,
  - Marks it as hypothetical and reflects this in data confidence levels,
  - Saves his SITREP in his database,
  - Arms trigger(s) at the relevant subordinate MDCs to notify him when the missing SITREP arrives,
  - Adds the unit(s) with the late SITREP(s) to a check list that will prompt him to respond, each time he takes an action, as to whether the action is dependent on the hypothetical information.
- A corps KMC sends its SITREP to its commander (e.g., Army HQ or joint task force).
- All KMCs prepare a message containing the SITREP and any related database triggers, check lists, etc., including instructions to the MDC whose local KMC is its commander to deliver the SITREP to the local KMC, and sends the message to all its MDCs.

At the battalion-, brigade-, division-, corps-level MDC:

- The message is received from the KMC and the MDC database is updated to contain the message data.
- If the MDC's local KMC is its commander, then the message is forwarded to the local KMC.

Let us suppose a brigade SITREP is missing, and its hypothetical SITREP is added to a check list. Each time the division commander/staff causes an action (e.g., a message is sent, a disk is written, a report is written, or a decision process is used), the commander/staff is asked to look at the check list and mark any dependencies. If a dependency is checked, then information is recorded about the check list item and the action taken, enough to identify the recipient of the action and the type of action or product, so changes can be propagated to the recipient if necessary when the missing data arrive.

Let us suppose the node is a division node and the commander is missing a SITREP from one of his brigades. He requests the missing SITREP from his brigade MDC:

— The brigade MDC checks whether there are timely SITREPs for the battalions reporting to its brigade, and:
  — If so, it uses the battalion SITREPs to prepare its brigade SITREP,
    - Else it will ask each battalion MDC that is missing its most recent SITREP to estimate its SITREP and send it to the MDC, lists their SITREPs as hypothetical in its database, and uses them in preparation of its SITREP,
    - The brigade MDC arms trigger(s) for any battalion(s) with hypothetical SITREP(s) to notify it when their missing SITREP(s) arrive,
  — It marks its SITREP as hypothetical and reflects this in data confidence levels,
  — It saves its hypothetical SITREP in its database and sends it to its KMC.

When a battalion receives a late SITREP, the trigger is activated causing it to send its SITREP to the brigade MDC and to turn off the trigger. The brigade uses the SITREP to compute a new SITREP and compares it against a threshold to see if it has changed enough to be sent to the division KMC as an update to the previously estimated SITREP. If so, it is sent.

When the brigade MDC receives its delayed SITREP, it stores it in its database, and the trigger causes it to send the SITREP to the division KMC and turn off the trigger. The division KMC:

— Prepares an updated division SITREP.

— If there are no estimated brigade SITREPs pending update, the division sends its late SITREP to its MDCs and, via an MDC, to its commander or, in the case of the corps, sends a SITREP message directly to its commander.

— The differences between the previously estimated SITREP and the latest SITREP are compared and if they exceed thresholds, or if the commander/staff determine the differences are great enough, the dependency list is used to create messages to propagate the changes.

Example Three: Change in Operation Orders (OPORDs)

In this example, the Commander-in-Chief (CINC) develops an operation plan for accomplishing an assigned mission. The plan becomes more specific and focused as it is passed down the command chain. By order of the National Command Authority (NCA), the CINC will begin execution planning, which will result in turning the OPLAN into an OPORD that will include detailed operation orders developed by component commanders. The OPORDs represent the way the CINC and his component commanders intend to fight the war. The component commanders, in turn, send these OPORDs down to the Army corps commanders and so on down the hierarchy. The intent is to simulate, sand-table, and practice the plans at every command level to develop robust OPORDs that include pre-thought-out sub-plans (FRAGOs) to cover most probable contingencies. The FRAGOs make the OPORD more robust and flexible. If a contingency is recognized as a FRAGO, then the OPORD direction can be changed accordingly without major replanning.

Contributing to the Army OPORD is an analysis of the enemy threat, which includes detailed analyses of the different avenues of approach that the enemy may take based on existing roads and their passability, and terrain features such as elevation and vegetation. The intelligence preparation of the battlefield ascertains decision points (e.g., places at which the enemy might change direction) and obstacles the enemy must move through, such as bridges or narrow passes. These obstacles present opportunities for the friendly forces to act to disrupt or delay the enemy's plan. The OPORD is based on this type of analysis. The number and complexity of the FRAGOs depend heavily on such things as time available for in-depth analysis, the detail of intelligence, terrain, and weather information available, detail of understanding of enemy doctrine, and training and readiness of friendly forces in the particular situation. As OPORDs are being carried out, situation reassessment is continually being done to determine if a change to a FRAGO is needed or a more extensive change is needed that requires replanning and changing of the OPORD.

As OPORDs move down the C2 hierarchy, they become more explicit and detailed. OPORDs consist of a map or graphical presentation of the mission to be performed and a written description. The explicit action part of the description is written using a highly restrictive subset of English, developed to promote understanding and reduce ambiguity, and has, with human assistance, been parsed by computer programs.

A major concern of commanders at all levels is that their OPORDs are completely and unambiguously understood by their subordinate commanders. This is currently done by holding briefings in front of wall maps and using grease pencils on acetate overlays to indicate the plan. In the future, OPORDs could be "played" at different command levels by animating them on an electronic map with replicated underlying databases. The objects being shown would be representations of the objects in the databases, and, for example, subordinate OPORDs could be simulated and gamed to see if they adequately performed the mission.

Kahan, Worley, and Stasz (1989) present the commander's data needs in terms of his dynamic image of the battlefield, which includes his understanding of the history of the situation as well as his projected futures, which rest on his own and the enemy's possible actions. They identified three modes of information exchange between commanders and their staffs: the pipeline, the alarm, and the tree. The pipeline transmits information according to a set order and an established format; an example would be a SITREP. The alarm signals the occurrence of one or more exceptional events. Alarms can be explicitly set by commanders, e.g., an unexpected sighting of the enemy, or implicitly set by the commander's staff through an understanding of his plan and recognition of events that indicate a change. The tree mode is an inquiry-based, demand-pull means of searching for and acquiring information in response to specific demands that arise from previously supplied information such as through an alarm. When the commander's image agrees with the world information, he operates in pipeline and alarm mode; when his image is disturbed, he operates in tree mode to acquire information he did not previously know he would need. Pipelines trade off low timeliness and a set level of detail for a low degree of uncertainty. Alarms are quite timely and often quite specific. Trees carry information of varying timeliness and detail.

Our replicated database design is intended to increase the availability of tree-mode data. Recognizing the need to change the OPORD is an alarm mode of operation. For example, a position location change of a friendly unit that exceeded a threshold (as previously discussed) would trigger an alarm to examine whether this change would affect the OPORD.

We discuss two examples below, one showing an OPORD change moving from the platoon upward and the other an OPORD change coming into the corps from its superior commander and moving downward. In the bottom-up example, the middle linkage is left open, since it could work in several different ways depending on doctrine and agreed-to protocols. Doctrine or procedures may dictate that a unit cannot change its OPORD until commanded to do so from above; the change would have to move up through corps and then be passed down through battalion. It is most likely that something as important as an OPORD change will not take place entirely through the data distribution system but that once a critical situation is recognized, changes to a new FRAGO would be discussed and agreed to by commanders and staff communicating over CNR networks. New OPORDs would then be entered at the appropriate command levels where they would be sent by each KMC to all of their MDCs.

Our examples below assume that the OPORDs are in a machine-readable form and have defined threshold tests for determining when there is a change to a new FRAGO or a more serious change that requires replanning. Our first example is of an OPORD change originating at the platoon level.

At the platoon level:

— A platoon en route to a position has encountered mechanical problems with two tanks; repairs will delay the platoon so that it cannot carry out its current plan. A high-priority OPORD change request message is sent to the company commander.

— The OPORD change request is sent horizontally to other platoons, companies, battalions, etc., in the geographically relevant and mission-relevant area.

At the company level:

— The message is received from the platoon and receives high-priority processing.

— The message is entered in the company database and is evaluated with respect to the overall company ability to support the current OPORD.

The evaluation determines that the OPORD cannot be carried out as currently planned but a change to FRAGO X can be met.

— A high-priority OPORD change message is sent to the battalion commander requesting that the company change to FRAGO X; it includes the reasons for the change—disabled tanks and the estimated time to repair.

— The OPORD change request is sent horizontally to other companies, platoons, battalions, etc., in the geographically and mission-relevant area.

At the battalion-level KMC:

— The message is received from the company and receives high-priority processing.
— The message is entered in the battalion database and evaluated with respect to the battalion ability to support the current OPORD, or FRAGO X, and the battalion commander determines that FRAGO X is the best way to go.
— An OPORD change request message is sent to the KMC's MDCs at each command node. It includes:
  — Indication that this is a priority message that should be delivered to the MDC's local KMC immediately;
  — Battalion commander's evaluation and recommendation to go to battalion OPLAN FRAGO X.
  — The message received from the company includes the reasons for the change—disabled tanks and the estimated time to repair.

At the battalion-level MDC:

— The message is received from the KMC and the data are stored in the MDC database.
— It is delivered to the local KMC as a high-priority message.

At the battalion-, brigade-, division-, or corps-level KMC receiving an OPORD change (or command) message from a battalion, brigade, division, or corps MDC:

— The message is received from the MDC as a priority message and processed according to doctrine, protocol, or command-level structure. For example, if putting an OPORD change in effect is not a lower-echelon decision but must come from the top down as an OPORD command message, then:
  — The KMC, that is, the immediate brigade or division commander of the MDC affected, would evaluate the subordinate request for an OPORD change recommendation.
  — If it was unnecessary to change the OPORD at the KMC command level, then it would decide on the OPORD changes to be sent to its direct subordinates, enter these in its database, and send them as an OPORD command in a high-priority message to its MDCs.

— If it was necessary to change the OPORD at its command level, then it would develop its own OPORD change request and recommendation, store it in its database, and send it to its MDCs as a high-priority OPORD change request for forwarding to its commander.

— If the MDC is a division, then the corps KMC would evaluate the OPORD change request and would modify its OPORD or replan accordingly and then would send the new OPORDs for all of its brigades in an OPORD command message to all of its MDCs and it would notify its commander of its OPORD change.

Unlike an OPORD change message, an OPORD command message would be acted upon by KMCs who were direct subordinates of the MDC. The new OPORD would be made into more detailed OPORDs for the KMC's direct subordinates, stored in the KMC database, and sent in an OPORD command message to their MDCs.

The battalion receiving an OPORD command from its brigade would send new OPORD commands to its companies, and horizontally to other companies, platoons, battalions, etc., in the geographically relevant and mission-relevant area. Each company would send new OPORD commands to its platoon leaders, and horizontally to other companies, platoons, etc., in the geographically relevant and mission-relevant area. Finally, each platoon leader would send its plan to its tanks.

If an OPORD command change came into the corps from above, the corps would process it into detailed OPORD commands for its divisions, store the information in the database, and send it in a priority OPORD command message to its MDCs where it would be handled as described above for OPORD commands.

Although OPORD command messages contain OPORDs for all the direct subordinate units, protocol could specify that each subordinate KMC copy only its own OPORD from its commander into its database since its siblings' OPORDs would soon appear in their respective MDC databases.

Example Four: Handling of Intelligence Reports

The location and structure of enemy forces or intelligence information are one of the major information requirements for situational assessment and one of the most difficult to manage. They introduce issues that need to be accommodated, including security (especially protection of information sources); rapid dissemination of intelligence information; working with inconsistent, errorful,

maliciously misleading, and estimated data; working with data at multiple levels of detail; and working with data in multiple forms (e.g., voice reports, keyboard entry reports from the battlefield, images, and text documents).

As with a change in OPORD, an unexpected intelligence report will probably not be handled exclusively by means of the data distribution system. It is more likely that it will be discussed by intelligence staff at various levels through CNR and actions may be agreed upon and instigated at several levels at once.

Intelligence reports can enter the system from many different sources and at many different command levels: reports from human observers and interpreted imagery, the images themselves, sightings of enemy by reconnaissance planes, or sightings by friendly combat troops or by scout units. To accommodate all these different sources, the example design includes a Knowledge Manager for Intelligence (KMI) node that will have Mediators for Intelligence (MDIs) on every other node.

The first example shows an intelligence report moving up the command structure:

At the platoon level:

— A platoon en route to a position unexpectedly sights and engages an enemy tank unit.

— A priority intelligence message (or phone call) is sent to the company commander indicating location, direction, identification, approximate number, etc., of the enemy tank unit.

— The same message is sent horizontally to other platoons, companies, battalions, etc., in the geographically relevant and mission-relevant area.

At the company level:

— The message is received from the platoon, receives high-priority processing, and is entered into the company database.

— The priority intelligence message is immediately sent to its battalion and horizontally to other companies, platoons, battalions, etc., in the geographically relevant and mission-relevant area. (There may also be a direct voice or message notification to intelligence officers and thus to the KMI.)

— The company assesses the situation: (1) stores the assessment in its database, (2) may change the OPORDs to its other platoons to help the platoon engaging the enemy, (3) may develop a request help call to the battalion, and

(4) may construct an OPORD change request to battalion. Any or all of the above are packaged into an additional priority intelligence message and sent to the battalion.

At the battalion-level KMC:

— The message is received from the company and receives high-priority processing.

— The message is entered in the battalion database, and sent as an intelligence priority message to its MDCs. (Part of the MDC message processing could be to deliver the message to the KMI on the MDC's node.)

— The battalion assesses the situation using the local MDI: (1) stores the assessment in its database, (2) may change the OPORDs to its other companies to help the company engaging the enemy, (3) may develop a request help call to the brigade, (4) may construct an OPORD change request to brigade, and (5) may instruct the MDCs to forward the message to their local KMC. Any or all of the above are packaged into another priority intelligence message to send to its MDCs.

At the battalion-level MDC:

— The intelligence priority message is received and stored in its database and delivered to the local KMC for high-priority processing.

At the brigade and division KMCs:

— Commanders/intelligence staff at KMCs in the command chain of the MDC can assess the situation, augmenting data by calling for information from the local MDI: (1) the assessment is stored in the database, (2) the KMC may change the OPORDs for its subordinate units to help engage the enemy, (3) may develop a request help call to the next higher command level, (4) may construct an OPORD change request to the next higher level, and (5) may instruct the MDCs to forward the message to their local KMC. (1), (2), (3), and (5) are packaged into a priority intelligence message; (4) and (5) are packaged into an OPORD command message and both messages are sent together to the MDCs.

At the corps KMC:

— The corps commander/staff can assess the situation, augmenting data by calling for information from the local MDI: (1) the assessment is stored in the database, (2) new OPORDs may be generated for divisions in response to the

intelligence report, (3) a help request to the next-level commander may be generated, and (4) instructions are given to the MDCs to forward the message to their local KMC. The message is sent to the MDCs.

At the KMI:

— The MDC intelligence assessment (of the unit that sighted/engaged the enemy) is added to the database and used by the KMI intelligence staff with all the other intelligence information to develop an integrated intelligence assessment that may be stored in the KMI database in different forms at different security levels to be made available to KMCs at their request. All added (new) data are sent in a message to the KMI's MDIs.

The next example shows an intelligence report moving down the command chain:

At the corps KMC level:

— An intelligence report is received at the corps level from above or outside the corps, pertaining to enemy forces within the corps area. (If the report comes from the KMI, then it would be delivered through the corps MDI as a message to the KMC and would have been replicated on all other nodes in the same way, through the resident MDIs.)
— The corps KMC stores the intelligence report in its database. Depending on the urgency of the report, it may send a priority intelligence report to all its MDCs (if the report did not come via the node MDI), then do an assessment using the MDI database and send out another priority intelligence report or it may just send out one report after the assessment.
— If the intelligence report assessment results in changed OPORDs, it develops the changed OPORDs for its divisions and sends those out in an OPORD command message to its MDCs.

At the corps MDCs:

— The messages are saved in the database.
— The messages are forwarded to the local KMC.

At the division, brigade, battalion KMC:

— KMCs that are direct subordinates to the MDC will assess the intelligence message with the local MDI data, store the assessment in their database and send this as a priority intelligence message to their MDCs. If the priority

intelligence message affects their actions they will develop new OPORDs for their direct subordinates and send these in a priority OPORD command message. (The two messages can be packaged and sent together.)

— All other subordinate levels respond to the priority intelligence message by immediately assessing it, determining how it may affect their actions, and storing tentative new OPORD(s) in their database. For all command levels, all of the database additions are packaged and sent to MDCs as a nonpriority intelligence message. (The MDCs simply store the data in their databases and do not forward them to their KMCs.) Battalions also send these data including tentative new OPORDs to their company commanders as priority intelligence messages.

At the division, brigade, battalion MDCs:

— The message is stored in the MDC database.

— If the message is a priority intelligence message or a priority OPORD message, it is delivered to the local KMC.

At the company commander level:

— The message is stored in the company database.

— If the message is a priority intelligence message, an assessment is made as to how this information affects the OPORD, and if it requires a change, tentative OPORDs for the platoons are made and saved in the database and the message and tentative OPORDs are forwarded to the respective platoons.

— If the message is a priority OPORD command, then new OPORDs are created for the platoons and forwarded to the respective platoons.

Discussion of the Examples

The major purpose served by these examples is to demonstrate the limitations of a technical approach without a strong domain understanding. The questions and shortcomings raised by these examples need to be addressed by a team that understands Joint Force and Army doctrine, military science, operational requirements, future visions for battle management, and current and future directions for information technology, as well as other technologies that will affect the way battles and operations other than war will be conducted.

The first example, the platoon position location change, raises issues about the requirements and technical feasibility (including communications bandwidth

constraints) of distributing all data everywhere on the battlefield, especially to battalion and below. Questions to be asked include: What level of data resolution is needed by units with different functions and command levels? What is the area over which each level/functional unit needs information? How is the area of data distribution calculated and shared (especially as units move around the battlefield)? Could satellite communications help, and if so, how?

The second example, the distribution of situation reports, also raises issues about how widely to disperse information like a SITREP to company and platoon levels. It does show, rather effectively, the advantage of having replicated data at remote sites. It illustrates how a division, in developing its SITREP, can estimate a SITREP it finds missing from one of its brigades by using the the brigade battalions' SITREPs and other recent data about the brigade.

The third example, moving OPORDs upward to corps and downward from corps, brings up an interesting doctrine issue. In the past, data always flowed upward and then laterally and downward, and orders flowed downward from higher to lower echelon. If information technology is applied to replicating all data everywhere as quickly as possible, then lower units will be knowledgeable about situations in which their plans need to be changed before they will get new plans from above. If new doctrine is not developed to allow them to act on the available information, then how much benefit does the information give the warfighter? TRADOC PAM 525–5 says that the new vision will give the subordinates as much information as is given to commanders and that individual soldiers will be empowered for independent action. How will subordinate units decide on compatible order changes without coordination from above? Assuming that data and software are replicated everywhere, the approach taken here suggests one way that technology can offer help: namely, for a unit to be able to simulate what other units would do (horizontal, superior, and subordinate units) and make its OPLAN changes based on that information. Another issue brought to the surface in this example is,who needs to know about detailed OPLAN changes, and who needs to know about tentative or hypothetical OPLAN changes? As shown in the example, the hypothetical OPLAN changes, as well as the real changes, were replicated everywhere for availability but the hypotheticals were not brought to the remote node's immediate attention. Hypothetical unit plans could be very useful to another unit making a simulated guess as to how the first unit would react in an unforeseen situation.

The last example, handling intelligence information, brings up many issues and was included to make the reader aware of the fact that this is primarily a military doctrine and decision problem and we rather arbitrarily demonstrated a way this

might work. The carousel databases shown in TRADOC PAM 525-5 need to be further explored from a military standpoint to understand better the technical challenges in management and dissemination of their data.

3. Research Issues and Future Directions

In Section 2, we described a design approach to replicated distributed databases for robust command and control that allows decisionmakers near-real-time access to the best data available under all conditions. The system is similar to the asynchrony of the real world in that some portion of the replicated data in the system will most likely be inconsistent at any point in time. Having accepted asynchrony and data inconsistency as a given, the design approach is to understand how best to serve decisionmakers in this imperfect real world by helping them (1) to use estimation and inference techniques when data are missing, ambiguous, sparse, and errorful; (2) to keep track of the hypothetical (e.g., estimated or inferred) data and their relationship to decisions; (3) to recognize significant differences in data changes; and (4) to propagate such changes when appropriate. The approach is realistic in realizing that some actions are irreversible and would never have occurred or would have been carried out differently if more accurate data and a better situational awareness were possible. The goal is to minimize these undesirable outcomes. In fact, decisionmakers frequently have to trade off between waiting for better information upon which to make a decision—and possibly deciding too late to produce effective results—or not waiting for more information and making a poor decision. At the very least, having extensive histories allows commanders, staff, and others a retrospective view of why decisions were made the way they were and, perhaps, insights into improved ways of operating in the future.

Current DDBMS products are based on the view that consistency is a major requirement and only custom-built products for particular applications have been specialized to provide asynchronous distributed support. Since DoD is encouraging the use of commercial off-the-shelf products and computer products are beginning to be developed in a modular way so that different capabilities can be mixed and matched, it is imperative to try to define what the needs of a robust C2 system are, to encourage research, development, and product efforts by government, universities, and industry.

Section 2 gave some indication by example of how a robust C2 system would work, but the examples were not complete and some of the mechanisms were alluded to without being well defined. In this section we will try to better define what is needed.

The major categories of capabilities needed are:

- Object-oriented technology combined with historical data management,
- Replication through use of knowledge-based object managers for C2 and other functional areas (intelligence, logistics, etc.) and their mediators for handling replicated data,
- Handling of local node long transactions (e.g., message processing, support for complex decisionmaking),
- Inference, estimation, temporal logic, and reasoning, and
- Handling of intelligence data and security issues.

Object-Oriented Technology Combined with Historical Data Management

The technology that seems most appropriate to the proposed robust C2 system design is object-oriented database technology that supports class hierarchies, multiple inheritance, multiple types of relationships, methods, and encapsulation coupled with support for managing historical data.

Object-oriented approaches provide a taxonomy for organizing objects and their relationships to each other. The popular object-oriented class hierarchies are based on a class/subclass or generalization/specialization relationship where a class or parent shares its more generalized concepts and methods with its subclasses or children through an inheritance relationship. A child class can then further specialize the concepts or methods or add additional ones, which can, in turn, be inherited by its children. Multiple inheritance allows a child to inherit concepts and methods from more than one parent, and may require conflict resolution if different parents offer similar but conflicting concepts and methods. Some object-oriented DBMSs (OODBMS) offer additional built-in relationships such as part/whole, where the child objects are related to the parent through functional roles.

C2 systems have many complex relationships such as command, control, support, and communication, which may each require different kinds of taxonomies implying different implicit functional relationships between the unit objects. What needs to be further researched is how complex these taxonomies are and how many and what kind are needed. This may require horizontal relationships between C2 functional areas and a force-level control system at each command level as well as special relationships between a command level and other Services. In addition, there are command, control, support, and other

relationships between a functional area and its systems at different command levels (e.g., the maneuver control system at corps, brigade, division, and battalion).

All this implies that an OODBMS is needed that allows definition of new relationships that are supported by the OODBMS similarly to the built-in relationships of class/subclass and part/whole.

OODBMSs encapsulate data structures and methods or programs that manipulate the data structures within an object module. Objects communicate with the external world through a well-defined interface that specifies a set of operations that can be performed on the instance objects defined by the object class. Encapsulation enables the exchange of well-defined information while hiding its underlying implementation. An object-oriented approach appears to be a convenient way to encapsulate the data and software methods used to perform data and knowledge management and decision support at each individual command node. The concept of encapsulation fits well with the design to replicate command objects as MDCs at multiple command nodes. One problem, however, is that the DoD Data Standards program is based on relational DBMS technology. If an object-oriented approach is used, it will be necessary to address how it will participate in and use the DoD data standards.

A robust C2 system requires the use and maintenance of historical data. Data are not replaced or changed in the system, rather they are added with a time stamp and, if necessary, other descriptive metadata such as their source, confidence of belief, method used to calculate them, etc. Thus, each object's attribute's values will have metadata-attribute-values associated with them, and an object-attribute-value and its metadata constitute one entry in a set of entries describing the way the attribute changed over time. These can be "originating" values reported from sensors or human sightings but most often are values derived through some type of processing, such as algorithmic, estimation, or inferencing.

Replication Through Use of Knowledge Managers and Mediators for Command and Control

The examples in Section 2 gave an intuitive feel for how the KMCs and MDCs would operate but more attention needs to be paid to the types of data in the KMC database that must be sent to all MDCs, and those that may not need to be sent to MDCs or at least not to all MDCs. In a strict sense, all changes could be sent to all nodes but if some are unnecessary, then not sending them will reduce network and processing loads. The issue mainly has to do with the sending of "control" data.

Future Decision Support Systems will be incorporating knowledge-based techniques and representations that could be envisioned as part of the OODBMS. An important part of future DSSs will be the ability for a commander or his staff to specialize the DSS to better fit a personal way of commanding or operating. In addition to being able to specify operating profiles, there will be ways for the commander and his staff to affect object methods by changing parameters (within specific limits), e.g., for threshold values, or by selecting rules or rulesets to be used in certain situations.

Any kind of operational change to KMC thresholds, parameters, rules, etc., that would affect an MDC's ability to estimate or infer KMC decisions must be sent to all the MDCs. How parameter and rule changes will be handled will need further investigation. It may be necessary to keep a history of such changes so that MDC results may be understood in terms of the status of their methods at the time they provided estimations or inferences.

As was said above, KMC input and output message logs must be maintained at all MDCs. Some of the controlling information (such as message acks or message sequence number lists) may not need to be sent to every MDC but only to those designated as possible backups or surrogates or identified as a sanctuary MDC. These MDCs could take over the KMC responsibilities at some future time and thus would need to have all the necessary controlling information to do so.

On the other hand, there will be parameter settings and rules for governing the interaction of the user(s) with the DSS, such as window settings, map resolution, colors, size, etc., that may frequently be changed interactively during operation. A design question is whether these kinds of data changes need to be handled as new data entries (historically handled) or in the conventional way as a change to an existing data value. If these are treated as historical data, then another issue is whether to send them as new entries to the MDCs. Although it seems unnecessary to send this type of data, the data could conceivably affect how a KMC would react in some situations, e.g., perhaps the alarm window is inadvertently hidden in a particular configuration, resulting in the user not reacting to an alarm as quickly as would be inferred. In fact, the settings of system parameters and the choice of activities or response to stimuli may represent distinct command styles that may be recognizable in an intuitive way but may be too complex to capture and communicate to the MDCs. Even if one were to capture them, the user interacting with the MDC methods would be part of the command staff at the local node command level and would have his/her own way or profile for interacting with the system. To be able to simulate the way another commander/staff would use the DSS would require very intensive human modeling.

Handling of Local Node Long Transactions

To help in understanding this subsection, the reader is referred to the appendix for a discussion of transaction processing and concurrency control.

A real concern, as discussed above, is to decide on the type of structure and support for local transactions including when and how they can be interrupted to handle new high-priority data. There are several conflicting needs. The new data need to be replicated at the remote nodes as quickly as possible but with minimal use of communications. The commander must complete decisions rapidly but wants the decisions to reflect up-to-date data that are constantly arriving. Is the commander best served by treating an entire decisionmaking session or all the actions resulting from an incoming message as a single, long, well-defined or ad hoc transaction, or is he better served by breaking up the actions into individual smaller transactions?

Suppose decisionmaking is treated as a long transaction. Then one needs to consider how other concurrent transactions may enter data into the database for possible use by the long transaction (or alternatively, could interfere with work already done on the long transaction). Other transactions might also begin processing high-priority messages at least to the point of making the decisionmaker aware of their presence. The appearance of a high-priority message may require that the decisionmaker switch his attention to the more critical matter. Does this just interrupt or should it abort the current transaction? Perhaps a KMC is processing an estimated SITREP when an update to some of the estimated data arrives. Should it abort the current transaction and start anew? If this happens frequently, there could be a lot of thrashing and a very long transaction if the data requirements trigger propagations.

One interesting aspect of the current approach is that it lends itself to the use of OODBMS technology, in particular, object encapsulation. This feature could offer support for many noninterfering concurrent data insert transactions run on behalf of the MDCs. One could speculate about multiple processor shared memory designs, or a fiber optic Local Area Network (LAN) constituting a node where each MDC is a subnode on the network.

Inference, Estimation, Temporal Logic, and Reasoning

One can see from the discussion in Section 2 that the KMC message input and output message logs, transactions, and metadata will contain information that can support backward and forward linking or reasoning about data derivations or events in the database. One could go backward from data in a database (1) by

using its derivation source and the transaction that entered it, (2) by linking from the transaction to an input message that contained the source data, (3) by locating the data in the "source" database using the time the message was sent, and (4) by using those data's derivation source and the transaction that entered them to link to the input message that contained the source data.

One could go forward starting from a data entry (1) by linking to an output message containing the data and their recipient, (2) by linking the output message identifier to the recipient's input message log, (3) by looking up the transaction that processed the input message, (4) by locating the new data entered by the transaction (it should reference the "message sending node" as a data derivation source), and (5) by linking to an output message (if any) containing the data and their recipient.

One could simulate and animate what actually happened when estimated or inferred data were used by following the data trail through the system, and then one could resimulate using the corrected data and look at the differences. The same simulation techniques could be used in decisionmaking if different estimation or inference techniques produced different outputs, to select the most robust interpretation.

The use and interpretation of thresholds, triggers, and propagation decisions need to be supported by tools that incorporate the concept of utility and meaningfulness of data to the overall process. The DSS must help the user understand the CCIRs in terms of the current situation and plans by providing decision support tools that help him determine what is important and how to express thresholds in terms of fixed values or a rule or ruleset.

In cases where a unit's position, SITREP, logistics needs, etc., have to be inferred because of interrupted communications, the reasoning tools should make use of all the relevant data available on that node. This includes data from units in the same general vicinity as the unit in question, the KMI data about the general intelligence picture and particularly the intelligence picture in the vicinity of the unit, more timely reports from the unit's subordinates or siblings, and the unit's OPORD and the OPORDs of its subordinates. Again, simulation can be used to try to determine what the range of possible positions and states of the unit could be. This is where the robust C2 design should really pay off. In more conventional designs, this type of information has to be pulled from other sources. Since the need for such data usually occurs at critical times, it is highly likely that the communication channels will be overloaded or unavailable—thus, the data arrive too late to be used effectively.

Handling of Intelligence Data and Security Issues

The need to create the KMI and MDIs in the design was driven by the need to account for the handling of intelligence information without a good understanding of what the options are.

As was said above, intelligence reports can originate from many different sources, such as human observers behind the enemy lines, voice/data reports from battlefield units and scout units, human sightings and images from reconnaissance flyovers, imagery interpretation reports, ground sensors, reports from coalition forces and other service components, and DIA reports and orders of battle. Intelligence can be in many forms: voice, electronic data, hard copy text, and images. The information can be sparse, conflicting (sometimes maliciously so as to mislead), errorful (coordinates reported incorrectly), and at various levels of detail and confidence.

It is generally agreed that it takes expert analysts to integrate and make coherent sense out of the wide array of information from the various sources. It is also generally agreed that it usually takes a long time for critical exploited intelligence information to get to the tactical user that needs it. To add to the problems, there are also very critical security concerns that may be addressed in ways that can affect other inferencing mechanisms. For example, security concerns often require that the source of the data be protected to the extent that the data might be modified to make it appear to come from a different source. Decision support systems often use or create metadata about domain data values that include confidence of belief levels for the data based on combinations of the sources and methods used in collecting the data as well as the number of independent sources reporting similar or conflicting data. If the data have been deliberately tampered with by the friendly side, then the inferencing techniques, say in a tactical DSS, will not work correctly.

Also for security reasons, all data may not be available to all levels of commanders, or the data may be available only if they have been modified, or they may never be available at certain nodes that may possibly be overrun by the enemy. Critical intelligence data may also be transmitted over secure telephone lines but will not be able to be entered into the database because of the security level, though they will be used by commanders in making decisions. So some judgments and decisions may be made without the DSS being able to trace them to source data.

For all of these reasons, we created the KMI and the MDIs. We reasoned that the KMI and its surrogates or backups would be behind the lines in safe locations or

in a sanctuary and that whoever in the intelligence chain commanded the KMI would have a multilevel multicompartmented system available and would designate the intelligence data available to each command node. The MDIs would not be fully replicated but would be partially replicated so that on each node at least that portion of the intelligence database appropriate to the KMC would be available and as much else as appropriate to other commands would be available for use in estimating and inferencing.

We also took into account that some intelligence reports would originate with tactical units (e.g., a platoon sighting and engaging an enemy force) and be passed up the command chain and once it reached the battalion level, would be sent to the battalion MDC on the KMI node for forwarding to the KMI.

We recognize that the design to handle intelligence data and secure data is still in a premature state. We recommend that work be conducted in this area to strengthen the design of these capabilities.

Issues for Future Exploration and Development

Military Issues

1. Army requirements/doctrine and technical concepts: Address how to bring together Joint Force and Army current and future doctrine, operational requirements, and future visions for battle management to strongly interact with and drive proposed technological solutions. This includes understanding Army C2 needs and relationships in detail.
   - — Address the resolution level of data needed by units with different functions and at different command levels.
   - — For battalion and below: How should the geographic area of data dissemination be computed? Frequency of computation? Exchange of such information? Address use of satellite communications.
2. Address concept of "switchboard in the sky," and broadcast channels for carousels of data shown in TRADOC PAM 525–5.
   - — Define differences/similarities between Force XXI "carousel databases" and C2 replicated data concepts.
   - — In particular, address intelligence and logistics.
3. Explore doctrine to support ability for units to change their actions based on current situational assessment without/before receiving orders from their commanding unit.

4. Explore the effect of voice communications: If critical information is exchanged only by voice and not entered into the data system, then inferencing techniques including simulation will be limited.
5. Security issues to address: (1) risk of sensitive information at battalion and below, (2) data/knowledge fusion problems dealing with classified data that have been downgraded to protect source, and (3) changes in security level due to data aggregation (replicating all data everywhere may introduce higher risk).

Technical Issues

1. Extending OODBMS technology: (1) to support application-defined relationships as first-class relationships (similar to support for class/subclass and part/whole relationships), (2) to represent and manipulate historical datasets, and (3) to develop adequate Army taxonomy that will support multiple views to organize data structure.
2. Local transactions: Better understand the types or classes of local transactions that need to be supported, investigate research in long transactions and complex transactions, and propose solution.
3. Synchronization of data through propagation of critical changes: Conceptualize decision support system tools for (1) the use of hypothetical data in decisionmaking, (2) the use of a checklist in linking hypothetical data use to actions, (3) further research into capturing the utility of data for checklist use, and (4) more detailed attention to how to propagate change.
4. Simulation of other node behavior: More research into simulation tools for allowing commander/staff on one node to simulate behavior of another node when the commander needs to (1) understand what the other node might do in a situation, (2) estimate the other node's capabilities.
5. Explore categorization of types of data: Is there a benefit to classifying types of data, and if so, then classify: types of metadata, hypothetical data, dynamic data, static data, etc?
6. Further explore use of DoD data standards with respect to definition and representation of data elements in OODBMS and historical DBMS.

Another issue is to understand the degree of compliance and compatibility of database schemas with the DoD data model. Are they required to be a derived view? What happens with local data not shared in the DoD Data Model? Lastly, there is a strong need to explore the use of self-defining messages based on DoD

data standards, a single message syntax to describe all messages, and one integrated message system to handle all C2 data (data, voice, graphics, images, video, email, etc.).

7. Further explore robustness: the use of replicated data and software to be able to support commander on other nodes, make command post movement transparent, and gracefully downgrade a battlefield system of systems suffering from hostile actions.

Appendix

Transaction Processing and Concurrency Control

In a multiuser DBMS environment, user programs may be reading and writing the database concurrently, operating in a interleaved manner. This means that operations from one program can execute in between operations from another program. Without proper control, interleaving could leave the database in an inconsistent state.

A database transaction can be thought of as a program, often delimited by "begin" and "end" statements, of database reads and writes and computation steps. It has four properties, often called the acidity of transactions: atomicity, consistency, isolation, and durability. Atomicity means that the transaction is always treated as a unit of operation. Consistency refers to its correctness in mapping one consistent state of a database to another consistent state. Isolation means that each transaction, during its operation, will see a consistent database at all times. And durability means that once a transaction is committed, its results are permanent. [Ozsu and Valduriez, 1991.]

Transactions can also be classified according to their structure. A flat or simple transaction has a single start point, body, and end. A nested transaction is a transaction that contains other transactions (subtransactions) each with its own begin and commit points. Advantages of nested transactions are: They provide a higher level of concurrency among subtransactions; it is possible to recover independently from failures of each transaction; and it is possible to create new transactions from existing ones by inserting the old one inside of the new one as a subtransaction. [Ozsu and Valduriez, 1991.]

"Concurrency control is the activity of coordinating the actions of processes that operate in parallel, access shared data, and therefore potentially interfere with each other" [Bernstein et al., 1987, p. 1]. If transaction processes were not allowed to interleave, then each transaction would operate one at a time, in a serial fashion. The goal of concurrency control methods is to develop transaction schedules that support interleaving of transaction processes and still achieve serializability.

There are two main concurrency control methods: locking of data entities, and timestamp ordering (TO) of transactions. Two-phase locking (2PL) is the technique of choice in most commercial DBMSs. It prevents interference by not permitting a transaction to request a lock after it has released a lock it already holds. There are many variations of locking algorithms; conservative 2PL requires a transaction to obtain all of its locks before performing any operations and strict 2PL requires the transaction to release all of its locks together when the transaction terminates [Bernstein et al., 1987]. There are variations of 2PL for replicated DBMS environments: In centralized or primary site 2PL, lock management responsibility is delegated to a single site; in primary copy 2PL, lock managers are implemented at a number of sites, each lock manager being responsible for managing the locks for a given set of lock units; and distributed 2PL expects availability of lock managers at each site [Ozsu and Valduriez, 1991.]

Timestamp-based concurrency control maintains serializability not through mutual exclusion but rather by an a priori transaction serializability order based on assigning each transaction a unique, monotonically increasing timestamp. The basic TO rule is: Given two conflicting operations belonging to different transactions, the operation belonging to the older transaction (earliest in time) is executed first. However, for the scheduler to detect operations that arrive out of sequence, each data item is assigned two timestamps: a read timestamp that is the largest timestamp of the transactions that have read it and a write timestamp that is the largest timestamp of the transactions that have written it. Transactions that are aborted are assigned new timestamps and restarted. Variations on TO are: a conservative TO that operates by delaying younger transactions until older ones can run; and multiversion TO in which updates do not modify the database but instead each write operation creates a new version of the data item marked with the timestamp of the transaction that creates it. [Ozsu and Valduriez, 1991.]

Pessimistic concurrency control techniques operate as described above and do not allow interference, and optimistic techniques detect interference at transaction commit time and then back out one of the transactions. Pessimistic techniques assume the application will have many conflicts, optimistic techniques assume the opposite.

We mentioned above that the concurrency control problem is different for a replicated database than for a single site database because of the need for consistency of the multiple copies. One copy serializability is required so that a user can access the same data from any copy. To achieve one copy serializability requires that distributed transaction schedules maintain mutual consistency. To do that necessitates synchronization among distributed sites, requiring message exchanges that decrease the timely availability of data. Because of the C2

application need for timely data, our approach traded off distributed data consistency in favor of data timeliness and availability with convergence toward consistency.

A large concern is transaction processing on the local command node. The command node will support multiple users performing many activities including maintaining status data, situational awareness and assessment, planning and replanning, issuing OPORDs, etc. Many of the activities will be accessing data and creating new data entries. Though we do not have a good application definition of what types of transactions need to be supported, we do know that some types of decisionmaking could conceivably be treated as long transactions, covering several hours. If those are treated as transactions, then during their operation, new data and information will arrive that need to be immediately handled and may, in effect, undo some of the work the transaction had already accomplished. Below, we review some research efforts that address similar problems or that may offer some insights that could be applied to the C2 solution.

Directly applicable to long transactions is Garcia-Molina and Salem's work, in which they define a saga as a long-lived transaction that can be written as a sequence of transactions that can be interleaved with other transactions. The DBMS would guarantee that either all the transactions in a saga successfully complete or that compensating transactions (i.e., furnished by the programmer) be run to amend a partial execution. A saga is similar to a nested transaction that permits only two levels of nesting: the higher, saga level and simple transactions. At the saga level, atomicity is not enforced, which means that at the saga level, a saga may view partial results of other sagas. The saga model could also be extended to include parallel transactions. Some insights into designing saga transactions would be to look for natural subdivisions of the work being performed or for cases in which the database is naturally partitioned into relatively independent components and actions affecting each component could be grouped as a saga transaction. Both of these suggestions could fit well with the C2 application. They go on to say that compensating transactions are difficult because they may have caused real-world actions that are hard to undo. The C2 design includes propagating changes to actions having a dependency on data changes, which is quite similar to the saga compensating notion. The authors suggest that a saga processing mechanism could be implemented with little effort as an added-on facility to an existing DBMS. [Garcia-Molina and Salem, 1987.]

Dayal, Hsu, and Ladin suggest the use of triggers and transactions to specify and organize long-running activities that involve multiple steps of processing. They

developed a model based on event-condition-action (ECA) rules and coupling modes whose execution is governed by an extended nested transaction model. They find sagas lacking because they are based on explicitly expressed control flow. They claim that ECA rules offer more flexibility in that events or situations of interest can automatically trigger additional activities. The approach also allows a more modular specification of the control structure including rule-based definitions of integrity constraints, exception conditions, etc., that are separate from the application steps. The ECA is an extended nested transaction model; if a triggering event occurs within a subtransaction of the top transaction, it generates a new subtransaction. Triggers can also signal from outside a transaction, in which case they would trigger the start of a new top-level transaction. Extensions also include the creation of deferred subtransactions, allowing a top transaction to be started from another transaction, and allowing a causally dependent-top-transaction to be spawned from inside another transaction. [Dayal, Hsu, and Ladin, 1990.]

Katz described a unified framework for version modeling in engineering databases. His emphasis is on the concepts suitable for structuring a database of complex engineering artifacts that evolve across multiple representations and over time and the operations through which such artifact descriptions are created and modified. He examined many different CAD models looking for unifying concepts. There are some similarities between the versioning needs of engineering databases and the C2 design proposed here: C2, like CAD, is a nontraditional application domain not satisfied by current DBMSs; current data are not overwritten with new data but maintained as historical data; and the data organization must handle multirepresentational and hierarchical data aggregates. Since CAD software is a multibillion dollar industry driving the extension of conventional DBMSs and C2 is not, it would be wise to determine if the CAD needs could satisfy C2 needs also. Some of the similar concepts include component hierarchies based on IS-A-PART-OF (e.g., Battlefield Functional Area (BFA) relationships); version histories based on IS-DERIVED-FROM (C2 need to understand the version of data the decisionmaker was working from at any particular point in time or to trace back to where derived data came from); and change propagation. [Katz, 1990.]

References

[ASB, 1992] Army Science Board Summer Study, *Command and Control on the Move*, 1992.

[Bernstein and Goodman, 1985] Bernstein, P. A., and N. Goodman, "Serializability Theory for Replicated Databases," *Journal of Computer Systems Science*, 31(3): 355–374, December 1985.

[Bernstein et al., 1987] Bernstein, P. A., Vassos Hadzilacos, and Nathan Goodman, *Concurrency Control and Recovery in Database Systems*, Addison-Wesley Publishing Company, Reading, Massachusetts, 1987.

[Chamberlain, 1994] Chamberlain, Samuel C., *The Information Distribution Technology: IDT—An Overview*, U.S. Army Ballistic Research Laboratory, BRL-TR-3114, Aberdeen Proving Ground, Maryland, April 1994.

[Chamberlain, 1990] Chamberlain, Samuel C., *Information Distribution Technology Research Program (brief)*, U.S. Army Research Laboratory, Aberdeen Proving Ground, Maryland, July 1990.

[Dayal, Hsu, and Ladin, 1990] Dayal, U., M. Hsu, and R. Ladin, "Organizing Long-Running Activities and Transactions," *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, 1990.

[DoD 8320.1, 1991] DoD OASD(C3I), DoD 8320.1 "Department of Defense Data to Force XXI," *Army*, May 1994.

[DoD 8320.1-M, 1993] DoD OASD(C3I), DoD 8320.1-M "Data Administration Procedures (Draft)," 21 June 1993.

[DoD 8320.1-M-1, 1993] DoD OASD(C3I), DoD 8320.1-M-1, *Data Element Standardization Procedures*, January 1993.

[DoD 8320,1-M-x, 1993] DoD OASD(C3I), DoD 8320.1-M-x, "DoD Data Model Development, Approval, and Maintenance Procedures (Draft)," May 1993.

[FIPS 183, 1993] U.S. Department of Commerce Technology Administration, National Institute of Standard and Technology, *Integration Definition for Function Modeling (IDEF0)*, FIPS PUB 18321, December 1993.

[FIPS 184, 1993] U.S. Deptartment of Commerce Technology Administration, National Institute of Standard and Technology, *Integration Definition for Information Modeling (IDEF1X),* FIPS PUB 18421, December 1993.

[Garcia-Molina and Salem, 1987] Garcia-Molina, H., and K. Salem, "Sagas," *Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data,* 1987.

[IDA, 1994] IDA T-J1-1285, *Command and Control (C2) Core Data Model, Version 1,* 1 July 1994.

[Kahan, Worley, and Stasz, 1989] Kahan, James P., D. Robert Worley, and Cathleen M. Stasz, *Understanding Commanders' Information Needs,* RAND, R-3761-A, June 1989.

[Katz, 1990] Katz, Randy H., "Toward a Unified Framework for Version Modeling in Engineering Databases," *ACM Computing Surveys,* Vol. 22, No. 4, December 1990.

[Oder, 1994] Oder, Joseph E, "Digitizing the Battlefield: The Army's First Step," 1994.

[Ozsu and Valduriez, 1991] Ozsu, M. T., and Patrick Valduriez, *Principles of Distributed Database System,* Prentice Hall, Englewood, New Jersey, 1991.

[TRADOC PAM 525–5, 1994] U.S. Department of the Army, TRADOC Pamphlet 525–5, *Force XXI Operations, A Concept for the Evolution of Full-Dimensional Operations for the Strategic Army of the Early Twenty-First Century,* 1 August 1994.

[USA ABCS, 1994] U.S. Department of the Army, TRADOC, *Army Battle Command System (brief),* presented to Army Science Board Summer Study during summer 1994.

[USA CCIR, 1985] U.S. Department of the Army, Combined Arms Combat Development Activity, *Division Commander's Critical Information Requirements CCIR,* Ft. Leavenworth, Kansas, 4 April 1985.

[USA Enterprise, 1994] U.S Department of the Army, ODISC4, *Army Enterprise Strategy Implementation Plan,* August 1994.